INTERNATIONAL STANDARD

ISO 19440

Second edition
2020-05

Enterprise modelling and architecture — Constructs for enterprise modelling

Modélisation et architecture d'entreprise — Constructions pour la modélisation d'entreprise

Reference number
ISO 19440:2020(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 184, *Automation systems and integration*, Subcommittee SC 5, *Interoperability, integration, and architectures for enterprise systems and automation applications*.

This second edition cancels and replaces the first edition (ISO 19440:2007), which has been technically revised.

The main changes compared to the previous edition are as follows:

— updates to the terms and definitions to address latest practice and harmonize with ISO 15704 and ISO/IEC/IEEE 42010;

— reorganization of the material into four components (metamodel, core constructs, specializations and model views), as described in 6.1;

— separation of the constructs into a slightly smaller number of core constructs in 6.3 and Clause 7, and specializations thereof (which can be extended by a model user) in Clause 8;

— introduction of a Service construct as a core construct, with specializations in 8.3 to address servitization;

— expansion of the Decision View in Clause B.4 and introduction of a new Collaboration View in Clause B.5 to demonstrate extensibility, i.e. use in other application domains;

— renaming of the construct property 'descriptive' as 'attribute';

— introduction of a Role core construct as a generalization of Organizational Role, Operational Role and Person Profile;

— introduction of a Performance Indicator core construct to support operational monitoring and process improvement;

— allowing an Enterprise Activity to be decomposable into sub-activities;

— elimination of Functional Entity to reduce the number of core constructs and replacing it by 'active Resource';

— insertion (in Clause A.1) of new text explaining why and how behavioural rules need to be constrained;

— deletion of annexes which will be included in a future Technical Report on typical usages of identification and usage of constructs in each model phase.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

This document defines the generic concepts that are required to enable the creation of enterprise models for industrial and other businesses and to provide support for the use of frameworks by industrial and other enterprises. This document builds upon ISO 19439 and defines and details a set of conformant user-oriented modelling language constructs for manufacturing and related services, which provide common semantics and enable the unification of models developed by different stakeholders in the various phases of model development. Such models are aimed at model-based support of operational decision-making and can be employed for model-based operation monitoring and control.

The modelling language constructs defined in this document can be specialized or assembled or both specialized and assembled into structures for specific purposes, for example for an industry or enterprise sector or for a distinct kind of enterprise concern such as maintenance. In turn, such structures and generic modelling language constructs can be used for developing distinct models for a specific enterprise.

The general requirements that determine the characteristics of the core constructs necessary for computer-supported modelling of enterprises are

a) the provision of an explicit model of Business Processes, with their dynamics, functions, information, resources, relationships and organizational responsibilities,

b) sufficient detailing and qualification of enterprise components to allow the creation of a model for a specific enterprise,

c) support for management of change, and

d) end-user-oriented representation to enable operational use.

NOTE All Unified Modelling Language (UML) class model figures are computer-generated scalable vector graphics (SVG). All generalization-specialization relationships in those class models are coloured red for increased clarity. Figures B.10 and B.11 are line drawings.

The names of terms representing core constructs (see Clause 7) and derived constructs (see Clause 8) are capitalized throughout this document to aid the reader in distinguishing them from general usages of the same term, specifically in order to distinguish the constructs Capability, Domain, Enterprise Activity, Event and Resource from general usage of capability, domain (or enterprise domain), enterprise activity, event and resource.

Enterprise modelling and architecture — Constructs for enterprise modelling

IMPORTANT — The electronic file of this document contains colours which are considered to be useful for the correct understanding of the document. Users should therefore consider printing this document using a colour printer.

1 Scope

This document identifies and specifies constructs necessary for users that model enterprises in conformance with ISO 19439.

This document focuses on, but is not restricted to, engineering and the integration of manufacturing and related services in the enterprise. The constructs enable the description of structure and functioning of an enterprise for use in configuring or implementing in different application domains. This document specifies an implementation framework in Clause 6 to map model constructs into such domains.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
aggregation
<enterprise modelling> process of, or result of, combining *modelling language constructs* (3.50) and other *model* (3.49) *components* (3.9) into a *whole entity* (3.35)

Note 1 to entry: Modelling language constructs and other model components can be part of more than one aggregation.

Note 2 to entry: Both Part-of and Consists-of are used in the aggregation *relationships* (3.64) described in 7.16.

3.2
attribute
piece of information stating a property of an *entity* (3.35)

[SOURCE: ISO 19439:2006, 3.2]

3.3
behavioural rule
<enterprise modelling> description of the logical sequencing *relationships* (3.64) of constituent activities used in the specification of *Business Process* (3.4) and *Service* (3.71) behaviour

3.4
Business Process
<enterprise modelling> *construct* (3.12) that represents a partially ordered set of Business Processes, Services and Enterprise Activities that when executed, achieves some desired end-result in pursuit of a given objective of an *enterprise* (3.27) or a part of an enterprise

3.5
capability
<general> ability to perform a specified activity

3.6
Capability
<enterprise modelling> *specialization* (3.73) of the *Enterprise Object* (3.33) *construct* (3.12) that represents the collection of *capability* (3.5) characteristics [expressed as capability *attributes* (3.2)] of either a *Resource* (3.67) (its provided Capability) or an *Enterprise Activity* (3.28) (its required Capability)

3.7
class
abstraction representing and encapsulating properties, *relationships* (3.64) and behaviour, which distinguish a collection of similar phenomena

Note 1 to entry: Class is used in a very general sense without any connotation for implementation or for use with a specific methodology.

3.8
complementary concept
conceptual representation that is not itself a *construct* (3.12), but that has a distinct significance and semantics for the purposes of enterprise modelling

Note 1 to entry: Many *construct templates* (3.14) use complementary concepts, which are listed in 5.12.

3.9
component
<system> *entity* (3.35), with discrete structure within a *system* (3.75), which interacts with other components of the system, thereby contributing to the system properties and characteristics

[SOURCE: ISO 19439:2006, 3.6, modified – The words “at the lowest level” have been deleted after “contributing”.]

3.10
concept definition phase
enterprise model phase (3.31) that defines the business concepts of an *enterprise domain* (3.29) to be employed in realizing its business objectives and its operation, including the necessary enterprise domain inputs and outputs

[SOURCE: ISO 19439:2006, 3.7, modified – The word “phase” has been added to the term.]

3.11
constraint
restriction or limitation or condition placed upon a *system* (3.75) that originates from inside or outside the system under consideration

[SOURCE: ISO 19439:2006, 3.8]

3.12
construct
<enterprise modelling> abstraction devised as an element of a modelling language to represent a generic concept in the *Domain* (3.24)

Note 1 to entry: This document addresses the domain of manufacturing, related services and collaborating enterprises.

3.13
construct label
literal string defined for each *construct template* (3.14), identifying the kind of *construct* (3.12)

Note 1 to entry: Construct labels are listed in 4.2.

3.14
construct template
common structure that allows the identification and description of distinct *modelling language constructs* (3.50) and the assignment of their properties

3.15
Co-provider
<enterprise modelling> *derived construct* (3.22), a *specialization* (3.73) of *Role* (3.69), which represents a person or an organization associated with another in providing a *Service* (3.71)

3.16
core construct
<enterprise modelling> dominant *construct* (3.12) that is distinguished from normal usage of the term by capitalizing the first letter of each word

Note 1 to entry: Core constructs are explained in Clause 7.

EXAMPLE *Domain* (3.24); *Business Process* (3.4); *Enterprise Activity* (3.28); *Service* (3.71); *Event* (3.36); *Enterprise Object* (3.33); *Enterprise Object View* (3.34); *Organizational Unit* (3.55); *Decision Centre* (3.17); *Role* (3.69); *Product* (3.63); *Order* (3.53); *Capability* (3.6); *Performance Indicator* (3.59).

3.17
Decision Centre
<enterprise modelling> *specialization* (3.73) of the *Enterprise Object* (3.33) *construct* (3.12) that represents a set of decision-making activities that are characterized by having the same time horizon and planning period and belonging to the same kind of *decision function category* (3.18)

Note 1 to entry: The terminology used to describe aspects of Decision Centre is found in CEN/TS 14818, which defines (time) horizon as "the part of the future taken into account by a decision, i.e. the horizon is six months when a decision is taken on a time interval of six months" and (planning) period as "the time that passes between a decision and when this decision shall be re-evaluated".

3.18
decision function category
<enterprise modelling> set of decision activities or *Decision Centres* (3.17) handling the same kinds of decision-making activities and concerning the same kinds of subjects

EXAMPLE Manage resources; manage products; plan production.

3.19
declarative rule
set of objectives and *constraints* (3.11), possibly combined with a set of requirements expressed as text

Note 1 to entry: Declarative rules can be imposed on *Business Processes* (3.4) and *Services* (3.71).

3.20
decommission definition phase
enterprise model phase (3.31) that defines the final state of a decommissioned operational *system* (3.75), all its *components* (3.9) for a specific *enterprise domain* (3.29) and the processes employed to conduct the decommissioning, so enabling reuse or disposition of those components

[SOURCE: ISO 19439:2006, 3.11, modified – The word "phase" has been added to the term, and the word "particular" has been replaced with "specific" in the definition.]

3.21
derivation
<enterprise modelling> process of elaborating *enterprise models* (3.30) at successive *enterprise model phases* (3.31) from the *models* (3.49) established at preceding phases, reusing the available contents and extending them according to the needs expressed for a model phase

3.22
derived construct
<enterprise modelling> *construct* (3.12) that is specialized from an *Enterprise Object* (3.33) and that is distinguished from normal usage of the term by capitalizing the first letter of each word

Note 1 to entry: Derived constructs are explained in Clause 8.

EXAMPLE *Organizational Role* (3.54); *Operational Role* (3.52); *Person Profile* (3.61); *Stakeholder* (3.74); *User* (3.76); *Co-provider* (3.15); *Functionality* (3.40).

3.23
design specification phase
enterprise model phase (3.31) that specifies the *Business Processes* (3.4), together with *Enterprise Activities* (3.28) and rules, that are to be performed to achieve the requirements

[SOURCE: ISO 19439:2006, 3.13, modified – The word "phase" has been added to the term, the words "Business Processes" have been capitalized and the word "capabilities" has been replaced with "Enterprise Activities".]

3.24
Domain
<enterprise modelling> *construct* (3.12) that represents the portion of an *enterprise* (3.27) to be modelled providing for identification of the relevant information

3.25
domain identification phase
enterprise model phase (3.31) that identifies the *enterprise domain* (3.29) to be modelled with respect to its business objectives, the enterprise domain inputs and outputs and their respective origins and destinations

[SOURCE: ISO 19439:2006, 3.15, modified – The word "phase" has been added to the term.]

3.26
domain operation phase
enterprise model phase (3.31) that encompasses the operational use of the *Domain* (3.24) *model* (3.49)

[SOURCE: ISO 19439:2006, 3.16, modified – The word "phase" has been added to the term.]

3.27
enterprise
<enterprise modelling> human undertaking or venture that has definite *mission* (3.48), goals and objectives to offer products or *services* (3.70), or to achieve a desired project outcome or business outcome

3.28
Enterprise Activity
<enterprise modelling> *construct* (3.12) that represents all or some part of the most detailed extent of enterprise functionality required by *Business Process* (3.4) objectives, that defines the task or tasks to undertake and that identifies the inputs needed for its execution and the outputs created as a result

Note 1 to entry: The necessary inputs and *resources* (3.66) are identified in the Enterprise Activity template.

3.29
enterprise domain
part of the *enterprise* (3.27) with a given set of business objectives and *constraints* (3.11) for this *enterprise model* (3.30)

Note 1 to entry: In this document, "enterprise domain" is abbreviated to "domain" whenever it is used as a qualifier in such terms as "domain identification phase" and "domain model". Other usages of "domain" have the normal dictionary meaning.

3.30
enterprise model
representation of an *enterprise* (3.27) as well as *entities* (3.35) within an enterprise, their interrelationships, their decomposition and detailing to the extent necessary to convey what the enterprise intends to accomplish and how it operates

Note 1 to entry: An enterprise model, which is used to improve the effectiveness and efficiency of the enterprise, identifies and specifies essential *components* (3.9) and elements to any necessary extent of detail, including any subsystems and constituent *models* (3.49) of the enterprise, e.g. an enterprise architecture model.

[SOURCE: ISO 15704:2019, 3.6]

3.31
enterprise model phase
life cycle (3.45) phase of an *enterprise model* (3.30)

[SOURCE: ISO 19439:2006, 3.24]

3.32
enterprise model view
model view
view
selective perception or representation of an *enterprise model* (3.30) that emphasizes some distinct aspect and disregards others

[SOURCE: ISO 19439:2006, 3.25, modified – The additional terms "model view" and "view" have been added, and the word "particular" has been replaced with "distinct" in the definition.]

3.33
Enterprise Object
<enterprise modelling> *construct* (3.12) that represents information in the *enterprise* (3.27) describing a generalized or a real or an abstract *entity* (3.35) conceptualized as being a whole

Note 1 to entry: All other constructs in this document represent entities that have specific semantics requiring distinct properties and additional descriptions.

[SOURCE: ISO 19439:2006, 3.27, modified – The term "Enterprise Object" has been capitalized, the field "<enterprise modelling>" has been added, the words "piece of information in the enterprise domain that describes" have been replaced with "construct that represents information in the enterprise describing", the words "which can be" have been deleted before "conceptualized" and Note 1 to entry has been added.]

3.34
Enterprise Object View
Object View
<enterprise modelling> *construct* (3.12) that represents a collection of *attributes* (3.2) selected from an *Enterprise Object* (3.33) for some distinct purpose

Note 1 to entry: The collection is defined by a selection of attributes and possibly *constraints* (3.11) on those attributes.

3.35
entity
concrete or abstract thing in the *Domain* (3.24) under consideration

[SOURCE: ISO 15704:2019, 3.8]

3.36
Event
<enterprise modelling> *construct* (3.12) that represents a solicited or unsolicited fact indicating a state change in the *enterprise* (3.27) or its environment

Note 1 to entry: An event can be associated with an Object View containing information related to the Event.

3.37
exception
<enterprise modelling> *Event* (3.36) that is raised as the result of either an abnormal termination of a previously initiated *Business Process* (3.4), *Enterprise Activity* (3.28) or *Service* (3.71), or by a special external mechanism such as a watchdog timer

3.38
function view
<enterprise modelling> *enterprise model view* (3.32) that enables the representation and modification of the processes of the *enterprise* (3.27), their functionalities, behaviours, inputs and outputs

[SOURCE: ISO 19439:2006, 3.32, modified – The field "<enterprise modelling>" has been added.]

3.39
functional category
grouping of *entities* (3.35) for expression of a common purpose or *capability* (3.5)

3.40
Functionality
<enterprise modelling> *derived construct* (3.22), a *specialization* (3.73) of *Role* (3.69), which represents some aspect of what a product or *service* (3.70) can or cannot do for a *User* (3.76)

3.41
generalization
specific concept modified for a more general content, use or purpose, or act of removing or modifying detail from a specific concept to produce a generalization thereof

Note 1 to entry: Generalization is the inverse of *specialization* (3.73).

[SOURCE: ISO 19439:2006, 3.34]

3.42
implementation description phase
enterprise model phase (3.31) that describes the final set of processes, *resources* (3.66) and rules implemented to achieve the desired operational performance for execution of the *Business Processes* (3.4), *Services* (3.71) and *Enterprise Activities* (3.28) specified in the *design specification phase* (3.23)

[SOURCE: ISO 19439:2006, 3.38, modified – The word "phase" has been added to the term, the words "Business Processes" and "Enterprise Activities" have been capitalized and the word "Services" has been added.]

3.43
information view
<enterprise modelling> *enterprise model view* (3.32) that enables the representation and modification of the enterprise information as identified in the *function view* (3.38)

Note 1 to entry: The information view is articulated as a structure containing *Enterprise Objects* (3.33) that represent the information-related *entities* (3.35) of the enterprise [organization, *resources* (3.66) and information].

[SOURCE: ISO 19439:2006, 3.40, modified – The field "<enterprise modelling>" has been added and Note 1 to entry has been modified.]

3.44
integrity rule
statement in the *requirements definition phase* (3.65) concerning restrictions on information to ensure conformity to real-world reality

Note 1 to entry: Integrity rules are used to define restrictions in terms of *constraints* (3.11) on *attributes* (3.2) of *Enterprise Objects* (3.33).

3.45
life cycle
set of distinguishable *life cycle phases* (3.46) and steps within phases which an *entity* (3.35) goes through from its creation until it ceases to exist

[SOURCE: ISO 19439:2006, 3.42, modified – The word "phases" has been replaced with "life cycle phases".]

3.46
life cycle phase
distinguishable phases of development of an *entity* (3.35)

[SOURCE: ISO 19439:2006, 3.43, modified – The words "stage of development in the life cycle" have been replaced with "distinguishable phases of development".]

3.47
meta-model
description of modelling elements for use in constructing *models* (3.49) pertaining to a domain

Note 1 to entry: A meta-model is always relative to the model in which its modelling elements appear.

3.48
mission
<enterprise> characterization of the effect that an *enterprise* (3.27) expects to achieve through the fulfilment of functional requests for products or *services* (3.70)

3.49
model
abstract description of an *entity* (3.35) in any form (including mathematical, physical, symbolic, graphical, or descriptive) that presents certain aspects of that entity

[SOURCE: ISO 19439:2006, 3.47, modified – The words "abstract description of reality" have been replaced with "abstract description of an entity" and "presents a certain aspect" has been replaced with "presents certain aspects".]

3.50
modelling language construct
<enterprise modelling> textual or graphical part of a modelling language devised to represent, in an orderly way, the diverse information on common properties and elements of a collection of *distinct enterprise entities* (3.35)

3.51
occurrence
<enterprise modelling> single, actual realization of a *modelling language construct* (3.50) that represents a distinct *entity* (3.35) in the real world at the time the *model* (3.49) is used

3.52
Operational Role
<enterprise modelling> *derived construct* (3.22), a *specialization* (3.73) of *Role* (3.69), which represents a distinct set of operational tasks and the relevant human skills required to perform those operational tasks

3.53
Order
<enterprise modelling> *specialization* (3.73) of the *Enterprise Object* (3.33) *construct* (3.12) that represents all kinds of commissioning and control in the company or organization that are necessary to deliver a product to meet a customer, legal or environmental requirement

Note 1 to entry: This includes the information for order management/administration and planning and control of *Business Processes* (3.4) in an *enterprise* (3.27).

3.54
Organizational Role
<enterprise modelling> *derived construct* (3.22) of *Role* (3.69) that represents the distinct set of organizational tasks and the skill profile required to serve and fulfil the defined organizational responsibilities, consisting of a list of predefined or user-defined human organizational skills

3.55
Organizational Unit
<enterprise modelling> *specialization* (3.72) of the *Enterprise Object* (3.33) *construct* (3.12) that represents an *entity* (3.35) of the organizational structure of an *enterprise* (3.27), which is described by *attributes* (3.2) of the organization and *relationships* (3.64) to both lower and higher-level organizational entities

EXAMPLE Department; division.

3.56
organization view
<enterprise modelling> *enterprise model view* (3.32) that enables the representation and modification of the organizational and decisional structure of the *enterprise* (3.27) and the responsibilities and authorities of the persons and *Organizational Units* (3.55) and *Decision Centres* (3.17) within the enterprise

[SOURCE: ISO 19439:2006, 3.52, modified – The field "<enterprise modelling>" has been added and the words "individuals and organizational units" have been replaced with "persons and Organizational Units and Decision Centres".]

3.57
partial model
model (3.49) used as a reference model in a specific kind of industry segment or industrial activity

Note 1 to entry: A partial model is comprised of *modelling language constructs* (3.50) or other partial models. Partial models also enable a modeller to reuse already existing models built for other *Domains* (3.24).

[SOURCE: ISO 19439:2006, 3.54, modified – The words "specific type" have been replaced with "specific kind".]

3.58
performance indicator
<general> metric or measure for assessing the achievement of an objective

3.59
Performance Indicator
<enterprise modelling> *specialization* (3.73) of the *Enterprise Object* (3.33) *construct* (3.12) that represents a set of *performance indicators* (3.58) by which the completion of processes and the realization of goals can be assessed

3.60
perspective
<enterprise modelling> orientation of a stakeholder or *model* (3.49) user relative to an identified *Domain* (3.24)

Note 1 to entry: A stakeholder's orientation can be formed by stakeholder concerns, as well as by their training, experience, cultural background, and their motivations.

[SOURCE: ISO 15704:2019, 3.19, modified – The field "<enterprise modelling>" has been added.]

3.61
Person Profile
<enterprise modelling> *derived construct* (3.22), a *specialization* (3.73) of *Role* (3.69), which represents a set of personal skills and responsibilities essential for one or more of an *Organizational Role* (3.54) or an *Operational Role* (3.52), and that a person provides

Note 1 to entry: A Person Profile can be assigned to more than one person and conversely a person can fulfil more than one Person Profile for more than one *Organizational Unit* (3.55) or *Enterprise Activity* (3.28).

3.62
processable model
model (3.49) with specified syntax and semantics that can be processed by a computer for analysis, simulation or execution

3.63
Product
<enterprise modelling> *specialization* (3.73) of the *Enterprise Object* (3.33) *construct* (3.12) that represents the technical information for management/administration and for planning production of an enterprise product

Note 1 to entry: Relevant technical information can be product history, e.g. characteristics, versions or test and qualification data.

3.64
relationship
association between two or more *entities* (3.35) that is significant for some intended purpose

3.65
requirements definition phase
enterprise model phase (3.31) that defines the enterprise operations needed to achieve enterprise objectives and the conditions necessary to enable those operations, both being without reference to implementation options or implementation decisions

[SOURCE: ISO 19439:2006, 3.59, modified – The word "phase" has been added to the term, the words "operations needed" have been replaced with "enterprise operations needed".]

3.66
resource
<general> *enterprise entity* (3.35) that provides some or all of the *capabilities* (3.5) required to execute an *Enterprise Activity* (3.28)

Note 1 to entry: In this document, resource is used in the *system* (3.75) theory sense of entities that provide capabilities required by the system and are an essential part of the system itself. The resource description includes the identification and description of consumables (such as energy, air, coolant) that are required to be present in sufficient quantities to operate the resource and material process inputs that are required by the various activities (such as raw materials, parts and assemblies). These inputs are identified in the *function view* (3.38), described in the *information view* (3.43), and have the associated management responsibilities identified in the *organization view* (3.56).

3.67
Resource
<enterprise modelling> *specialization* (3.73) of the *Enterprise Object* (3.33) *construct* (3.12) that represents the provided *resources* (3.66) available to execute an *Enterprise Activity* (3.28)

Note 1 to entry: The Resource construct does not include human resources.

[SOURCE: ISO 19439:2006, 3.60, modified – The term has been capitalized, the field "<enterprise modelling>" has been added, the words "enterprise entity that provides some or all of the capabilities required to execute an enterprise activity" have been replaced with "specialization of the Enterprise Object construct that represents the provided resources available to execute an Enterprise Activity" and Note 1 to entry has been modified.]

3.68
resource view
<enterprise modelling> *enterprise model view* (3.32) that enables the representation and modification of enterprise *resources* (3.66) and roles

Note 1 to entry: The resource view is articulated as a structure containing *Enterprise Objects* (3.33) representing the set of resources required to execute enterprise operations.

[SOURCE: ISO 19439:2006, 3.61, modified – The field "<enterprise modelling>" has been added, the words "enterprise resources" have been replaced with "enterprise resources and roles" and Note 1 to entry has been added.]

3.69
Role
<enterprise modelling> *specialization* (3.73) of *Enterprise Object* (3.33) that represents the role of a person or thing in a specific context

3.70
service
<general> functionality resulting from interaction between a supplier (provider) and a *User* (3.76), often in the context of a supplied product in use by the User

3.71
Service
<enterprise modelling> *construct* (3.12) that represents a *service* (3.70)

3.72
servitization
process in a manufacturing *enterprise* (3.27) to augment products with *services* (3.70) to satisfy *User* (3.76) needs

3.73
specialization
general concept modified for a more limited extent, specific use or purpose, or the act of adding or modifying details to a general concept to produce a specialization thereof

Note 1 to entry: Specialization is the inverse of *generalization* (3.41).

Note 2 to entry: When referring to a *class* (3.7), specialization involves the construction of subclasses within a class for a distinct purpose, where the members of each subclass have one or more characteristics [*attributes* (3.2), *relationships* (3.64), behaviour or semantics] in common which are not shared by all other members of the class. When referring to a *model* (3.49), it refers to the progression from generic concepts to *partial models* (3.57) and particular models.

[SOURCE: ISO 19439:2006, 3.62, modified – Note 2 to entry has been added.]

3.74
Stakeholder
<enterprise modelling> *derived construct* (3.22) of *Role* (3.69) that represents an individual, team, *Organizational Unit* (3.55), or *class* (3.7) thereof, having concerns relative to their *perspective* (3.60) about an *enterprise* (3.27)

Note 1 to entry: Typical enterprise stakeholders include enterprise owners, enterprise customers, and enterprise employees responsible for receiving or delivering either products or *services* (3.70), and those persons or Organizations partnering with the enterprise to achieve its *mission* (3.48).

3.75
system
collection of items arranged and interacting for a given purpose

3.76
User
<enterprise modelling> *derived construct* (3.22), a *specialization* (3.73) of *Role* (3.70), which represents the interests and characteristics of an *entity* (3.35) using a product or *service* (3.70)

4 Abbreviated terms

4.1 General

EBNF	Extended Backus-Naur Form
GRAI	Graphes à Résultats et Activités Interreliés (Graphs of Interrelated Results and Activities)
HTTP	HyperText Transfer Protocol
ICT	Information and Communication Technology
UML	Unified Modelling Language
XML	eXtensible Markup Language

4.2 Construct-specific

BP	Business Process
BRS	Behavioural Rule Set
CA	Capability
CODM	Collaboration Domain
COPR	Co-provider
COPT	Collaboration Point
CPRT	Collaborating Partner
DA	Decision Activity
DC	Decision Centre
DECN	Decision
DFRM	Decision Frame

DM	Domain
DSTR	Decision Structure
EA	Enterprise Activity
EO	Enterprise Object
EV	Event
FUNC	Functionality
OPR	Operational Role
OR	Order
ORR	Organizational Role
OU	Organizational Unit
OV	Object View
OVPR	Object View Profile
PI	Performance Indicator
PPR	Person Profile
PR	Product
RE	Resource
RO	Role
SRV	Service
STK	Stakeholder
USER	User

5 Representations, relationships, roles and common concepts

5.1 Modelling language constructs, purpose and applicability

Clauses 7 and 8 define, describe and provide templates for generic modelling constructs. As a set, these constructs contribute to a modelling language for creating a wide range of Business Process-based enterprise models. When utilized, each construct shall include the attributes that are pre-defined in this document. Other attributes may be added to meet distinct user needs by a suitable extension of the template (see 8.2, 8.3, Clause B.4 and Clause B.5).

Constructs are often applicable to several model views and are elaborated to different extents at each of the modelling phases. Where the applicability of a construct, as defined in this document, is limited to a certain model view or enterprise model phase, that limitation is noted in remarks.

NOTE Because of the interrelationships and dependencies between constructs, they cannot be described in this document in strict linear sequence (i.e. with no forward reference to not-yet-described constructs). The sequence of text used herein for core constructs (and not intended as a modelling methodology) has been chosen to minimize this forward reference problem and is as follows:

— start with Domain (the scope and boundary of the model);

— add functionality in terms of Business Processes, Services, Enterprise Activities and Events;

— add information representation in terms of Enterprise Object and Object View, and the specializations Product and Order;

— provide for resourcing with Resource and Capability;

— lastly, address organizational and role issues as subclasses of Role, with Organizational Unit, Decision Centre, Organizational Role, Operational Role and Person Profile.

5.2 Dimensions of enterprise models

5.2.1 Dimension of genericity

Relative to the dimension of genericity defined in ISO 19439, constructs are generic elements for use in constructing partial and particular models. For partial models (e.g. industry or trade specific models), some attribute values may remain undefined for partial model instances (e.g. inputs/outputs for Events for Domains and inputs/outputs for Business Processes), with value definition occurring for the particular model instance. Such missing entries need to be completed for the particular model.

5.2.2 Dimension of enterprise model phase

The dimension of enterprise model phase in ISO 19439 addresses the life cycle of models and model components. The concern of this dimension is the development and evolution of the model of the domain, starting from the identification of the enterprise domain and progressing to a processable model and the decommissioning thereof. Therefore, the specifications of modelling language constructs need to accommodate their intended usage and representation in a distinct model phase. At each phase, all constructs specified at that phase shall be specified to a comparable level of detail. Attributes of modelling language constructs shall be adaptable and selectable for the different model phases according to the envisioned needs.

5.2.3 Dimension of enterprise model views

ISO 19439 and ISO 15704 use enterprise model views (often shortened to 'model views') to provide a selective perception of an enterprise that emphasizes some distinct aspect of the matter under consideration and disregards others, thereby reducing complexity of representation. Model views need to be understood as selective views on an integrated enterprise model. This document identifies four mandatory enterprise model views (Function, Information, Resource, Organization) that allow the modelling of the major aspects of an enterprise. Additionally, other views may be defined as needed and supported by the modelling methodology, e.g. economic view, decision view, purpose view and implementation view. In this case, additional attributes may augment the constructs defined in this document or relevant new constructs may be defined. Therefore, the specifications of modelling language constructs should accommodate their intended usage and representation in one or several model views. To assure consistency of constructs that can appear in more than one view, automated tools are usually necessary.

5.3 Construct representation

Modelling language constructs shall be represented in a form that is both understandable by humans and interpretable by digital technology. This means there is the need for

— a visual (graphical, iconic) representation of construct structures for use by model builders and business users,

— a structured information-capturing representation (for which this document uses templates) for use by model builders and business users, and

— a machine-interpretable representation for computer-based model simulation and model-based enterprise operation control and monitoring.

NOTE 1 Interpretability by digital technology is not the subject of this document but is provided by the modelling tool.

Representation shall be performed in the following four ways:

a) representation by a common structure of the constructs of this document, as specified in 5.4;

b) representation of construct properties by templates to arrange them into a common format, as specified in 5.5;

c) representation of the relationships between these modelling language constructs, as specified in the relationships section of each template and illustrated in the various Unified Modelling Language (UML) class models;

d) representation of dynamic behaviour by behavioural rules, as specified in 7.2.5 and Annex A.

NOTE 2 This document uses class diagrams as examples to illustrate constructs and the relationships between them.

5.4 Common structure for modelling language constructs

To describe a distinct modelling language construct, this document uses a common structure that provides a consistent minimal assignment of properties to each construct. This common structure shall have two constituents

a) a textual description consisting of brief text defining the modelling language construct in terms of a purpose, its description and its intended usage, and

b) a construct template, which arranges and defines the properties for this modelling language construct.

5.5 Template for modelling language constructs

The template is described in an informal manner but using a common form. The construct template shall have the form described below.

a) Header part having the same attributes for each modelling language construct and containing attributes relating to the identity of a construct and to its context in which it is used. It shall be structured as follows:

   1) construct label (a literal string denoting the kind of construct);

   2) identifier (a literal string that is unique for each occurrence of the modelling language construct within the model). The identifier consists of an uppercase construct label, followed by a user defined non-empty string of alphanumeric and separator characters followed by a space. The separator characters are restricted to full stop ('.'), dash ('-'), and underline ('_'), e.g. BP001, SRV.Ordering, EA-welding, OV1.2, RE_6786_A;

      NOTE 1 The characters solidus ('/') and comma (',') are not permitted in an identifier.

   3) name (the name of the instance of the modelling language construct);

   4) authority for model design (i.e. the identifier of the Organizational Unit and Organizational Role responsible for the design and modification of this construct).

   NOTE 2 The Manufacturing-oriented Enterprise specialization of 8.2 also includes Organizational Role and Person Profile (to identify the person currently assigned to an Organizational Role).

b) Body part containing the distinct properties that are specific to each construct and whose description is derived from the corresponding modelling language construct definition. Body parts shall be structured in two further partitions as follows:

   1) attributes, containing descriptive properties that comprise

      i) construct-identifying description in textual form,

      ii) construct attributes that are predefined,

      iii) additional construct attributes that may be defined by the user to meet distinct needs,

      iv) attribute qualifications – statements that are made about whether attribute values are mandatory or optional, when they are applicable, etc., and

      v) activation priorities – priority is used at run-time to select which process to respond to first when two processes complete within the same interval of observable time;

   2) relationships (further described in 5.10), containing relationship properties that can include

      i) Specialization-of relationships, representing relationships between a specialization and its generalization,

      ii) Part-of relationships, representing relationships between this construct instance and the whole aggregated from such instances,

      iii) Consists-of relationships, representing relationships between this construct instance and its constituent parts,

      iv) Association relationships for other forms of relationship, either predefined or user-defined (such as provision or usage of a Capability instance by Resource instances), and

      v) Operational relationships, representing the responsibility and authority for model operation (i.e. the identifier of the Organizational Unit and Operational Role responsible for the operational usage of this construct or authorized to change its usage).

NOTE 3 For Consists-of relationships and similar dual relationships (see 5.6), it is accepted that Part-of and Consists-of are complementary; depending on the purposes of the modeller, either or both can be used.

Where an attribute or relationship is not applicable for a certain enterprise model phase, that attribute or relationship is omitted in the construct template table instances.

Both attributes and relationships contain general properties and constraints on these and are applicable to enterprise model phases as indicated. Additionally, both attributes and relationships may be grouped as sets having some common characteristics, e.g. inputs/outputs or relationships to organizational concerns.

NOTE 4 In the models of Annex A, Part-of and Consists-of relationships are shown as UML aggregations without labels. UML associations are used to show Operational and Association relationships, and UML generalizations are used to show Specialization-of relationships.

NOTE 5 Version control has a significant impact on design and operation for enactable models.

5.6 Referential integrity

This document considers support for control of modelling language constraints to be a tool-provided responsibility. It is also the responsibility of the tool supporting model development and maintenance to ensure that attributes referred to in constraints and integrity rules are consistent with the attributes as defined for the construct instances involved.

When the value for a construct property is another construct instance, the modeller, and preferably the modelling tool, shall assure that the referred-to construct instance actually exists. This assurance

can be immediate as modelling occurs or as a distinct process to verify the referential integrity of model contents. In some situations, a reference to a construct instantiated in a later modelling life cycle phase can occur, but all such references need to be resolved to an existing construct instance before the modelling effort is complete.

Several relationships, e.g. 'Part-of construct' and 'Consists-of construct', are dual relationships. Where a construct specifies a dual-of relationship to another construct, depending on the purposes of the modeller, the other construct may specify the corresponding dual relationship – in that case, it is the responsibility of the modelling tool to ensure the consistency of these attributes.

NOTE The dual relationships used in the templates of this document are (where the uppercase codes are construct labels):

— assignedDC <–> controlsDC

— assignedORR <–> assignedToORR

— assignedOU <–> assignedToOU

— assignedToORR <–> assignedORR

— assignedToOU <–> assignedOU

— consistsOf <–> partOf

— controlsDC <–> assignedDC

— partOf <–> consistsOf

— providedCA <–> requiredCA

— providedInfo <–> requiredInfo

— relatedTo <–> relatedTo

— requiredCA <–> providedCA

— requiredInfo <–> providedInfo

— usedBy <–> uses

— usedByDC <–> usesDA

— uses <–> usedBy

— usesBP <–> usedByDM

— usesSRV <–> usedByDM

— whereProvided <–> whereRequired

— whereRequired <–> whereProvided

A modeller can specify other dual relationships, as required.

For constructs where both constraints and integrity attributes are used, it is accepted that these are also dual relationships; depending on the purposes of the modeller, either or both may be used. In the latter case, it is the responsibility of the modelling tool to maintain consistency.

For Event instances, it is the responsibility of the tool supporting model development and maintenance to ensure that the Event instances listed as Inputs or Outputs are compatible with the Generated-by attributes of Event. It is also necessary to ensure that the Generated-by and Initiates attributes are compatible with the Origin/Destination Event lists of Domain, and the Input/Output lists of Business Process and Enterprise Activity.

5.7 Representation of attributes

Each attribute shall be defined by its name and predefined data type (numbers, literals, strings, etc.).

NOTE 1 A data type can take on a range of permissible or actual values, such as a single identifier (id = "EO4711"), value range (diameter 10 mm ± 0,05 mm) or value list (colour is yellow, red or green).

NOTE 2 Constructs can use eXtensible Markup Language (XML) schema (see References [9], [10], [11]), EXPRESS (see ISO 10303-11) or other formal modelling specification languages as a means for representing attributes and the related definition of data types, expressions and statements.

5.8 Notation used to describe template contents

Properties are described with a combination of denotations and textual descriptions.

The notation used for this clause is that denotations enclosed in angle brackets are either self-explanatory (e.g. <integer>) or explained in accompanying text (e.g. <item>, where item is ...). NIL denotes a blank space. Literals in all upper case denote themselves as in DM, BP, EV, NIL, PHYSICAL, etc. The literal characters comma, forward slash, round brackets and equals are denoted by ':', '/', '(', ')' and '=' respectively.

Where more than one choice is possible for a denotation, these are separated by | (read as or), as in <a> | <b> | <c>. Square brackets '[' and ']' are used for convenience as a grouping container for one or more different items as in [<duration> <qual>] in a list of DURATION attributes of an Enterprise Activity, where <qual> is a code representing one of 'average', 'minimum', 'maximum', 'actual'.

Textual descriptions contain a brief definition of that property and, where appropriate, information providing additional details (shown in italics).

Multiplicities as part of textual descriptions are annotated by:

— [0..*] indicating no instances, a single instance or many instances,

— [1..1 | indicating a single instance,

— [1..*] indicating multiple instances or

— [m..n] indicating m up to n instances.

Comments are contained in a (*...*) bracket pair.

A non-empty list of <item> separated by commas is written as [<item>]+, while a similar possibly empty list is written as [<item>]*. An optional <item> is denoted as [<item>]?. Within a list denotation, <item> may be further decomposed as in

— [<id> <multiplicity>]+ or [<measure> | <metric>]*.

More formally, the syntax for lists is

— list = non-empty list | possibly-empty list;

— non-empty list = item | item, separator, non-empty list ; (* for convenience, abbreviated to [<item>]+ *)

— optional item = item | NIL ; (* for convenience, abbreviated to [<item>]? *)

— separator = ',' ;

— possibly-empty list = NIL | non-empty list ; (* for convenience, abbreviated to [<item>]* *)

EXAMPLE Examples of denotations are:

— From the Domain template (see 7.1.4), a Design Authority with values of Organizational Role and Organizational Unit would be expressed as [<identifier> '/' <name>] ':' [<identifier> '/' <name>]

— From the Capability template (see 7.14.4), RE_1006453/data_store would be valid for Where_provided <identifier> '/' <name>]+. (NIL would not be valid.)

— From the Enterprise Activity template (see 7.3.4), NIL or max_load = 10Kg; max_temp = 100ºC would be valid for '[<constraint>]* imposed on this Enterprise Activity'.

Some further examples follow where the leading uppercase codes refer to construct labels, followed by a property name, progressing from simpler to more complicated cases.

EO: Nature of Object PHYSICAL | INFORMATION.

CA: Included Capabilities [<Capability>]*

DM: Objectives [<objective>]+, strategic and operational business objectives of the Domain.

EV: Object Views [<origin> ':' <identifier> '/' <name>]* of Object View defining information associated with occurrences of this Event.

EV: Generated_by [<origin> ':' <identifier> '/' <name>]+ of the source of this Event.

BP: Object View Inputs [<origin> ':' <identifier> '/' <name>]+ of Object Views, occurrences of which can be received by a Business Process.

EA: Ending Statuses [<value> <priority>]+, ending status values produced by occurrences of this Enterprise Activity and their significance, where <value> is a mandatory 0-argument predicate and <priority> is an integer in a system-defined range representing the lowest and highest priorities respectively. By default, highest priority.

PR: Related-to [<identifier> '/' <name>]*, defining the Enterprise Objects that are related to this Product, where multiplicity [...]* is one of [0..*] (only for early modelling phases) or [1..1] or [1..n] or [m..n].

5.9 Support for modelling phases

This document aims at modelling of the enterprise supporting all enterprise model phases by modelling relevant language constructs as follows.

— At the domain identification phase, to identify the contents of the domain (its Business Processes) and its inputs and outputs, including Events, in such a way that the domain identification model can be used as the starting point for a derivation of a consistent concept definition model.

— At the concept definition phase, to define the business model of the enterprise with missions, strategies, etc., and in such a way that the concept definition model can be used as the starting point for a derivation of a consistent requirement definition model.

— At the requirements definition phase, to describe the Business Processes, Services and Enterprise Activities of the Domain from a business perspective, in such a way that the requirements definition model can be demonstrated to be sufficient for use as the starting point for the derivation of a consistent design specification model. The modelling process is eased by the availability of model Views (Function, Information, Resource and Organization – other views may be added if needed). These views are supported by a number of information objects (Enterprise, Resource, Organization) representing the many informational aspects of the enterprise operation. The requirements definition model shall also be capable of being verified by simulation.

— At the design specification phase, to specify the Enterprise Activities with all their components from both a business and Information and Communication Technology (ICT) perspective, in such a way that the design specification model can be demonstrated to be sufficient for use as the starting point for a consistent derivation of an implementation description model; therefore the design specification model shall also be capable of being processed for verification. In this phase, constructs shall be derived from those of the requirements definition phase by enriching them with properties that reflect all resources and general ICT interfaces.

— At the implementation description phase, constructs shall reflect the hardware/software platform or technology chosen to be consistent with design specification constructs. This means to describe the Business Processes, Services, and Enterprise Activities with all their components as they are

implemented in the actual system from both a business and ICT perspective, in such a way that the implementation model can be demonstrated to be sufficient for use in the operation of the enterprise for decision support and Business Process monitoring and control; therefore the implementation model shall also be capable of being processed for final verification and release for operational use.

— At the domain operation phase, the released model is used for operational purposes such as decision support, monitoring and control.

— At the decommission definition phase, to identify future use of the Business Process and Service components, in such a way that the decommission definition model can be used, as appropriate, as the starting point for reuse at any model phase.

However, this does not mean there will be a different set of modelling language constructs required for each model phase. Constructs are carried forward to subsequent model phases by providing an enhanced attribute list to capture additional needed information for representing and describing the enterprise entities at those model phases or by mapping onto different construct representations.

5.10 Representation of relationships

The purpose of a relationship is to model time-varying and other associations between run-time instances of modelling language constructs. Reflexive relationships shall be permitted, i.e. a relationship can be created between an instance and itself.

EXAMPLE 1 Relationships between entities can be those between a part and the machine which is used to produce it, or those between a worker and the production order to which he or she has been assigned, as in: Produced_by (part, machine), Works_at (worker, order). Such relationships are represented through assignments as inputs and outputs.

EXAMPLE 2 Illustrative relationships between constructs are those between a Domain and its Business Processes or those between a Business Process and its sub-processes and activities.

The concept of a user-defined relationship is dependent on the concept of Enterprise Object described in 7.6 and on the model entities derived from that Enterprise Object (Resource, Product, Order, Organizational Unit, Decision Centre). A user-defined relationship is used to describe domain-dependent relationships as perceived and defined by the users, and to describe the way in which they group structuring and behavioural constructs in terms of their utilization within the enterprise. User-defined relationships can be expressed by Related-to lists in Enterprise Objects, Products, Orders and Object Views on Enterprise Objects, or by additional user-defined properties.

5.11 Specializations

Depending on the purpose of modelling, it can be helpful and necessary for the modeller to define specializations of the constructs defined in this document. Additionally, this document predefines several normative specializations in Clause 8, Derived constructs.

Specialization can be achieved by

— specializing the most appropriate construct (and corresponding template) and adapting the header information accordingly,

— adding a new 'Specialization-of' relationship at the appropriate model phases(s),

— assigning appropriate values for the 'Specialization-of' property and listing the construct (or, exceptionally, constructs) of which it is a specialization,

— assigning or constraining values for existing properties as appropriate, and

— adding additional properties as required and assigning model-unique names for these.

Each construct can be further specialized into constructs of the same kind; for example, an Enterprise Object can be specialized into distinct kinds of Enterprise Object, Resource into further kinds of Resource, etc. A relationship 'Specialization-of' is introduced to hold these specialization relationships.

This document predefines several such specializations in Clause 7. Eight constructs (Capability, Decision Centre, Organizational Unit, Performance Indicator, Product, Order, Resource, and Role) are specializations of the Enterprise Object construct. These specializations can occur because the attributes and relationships of Enterprise Object exist whenever the object exists, even if some of those attributes and relationships are not shown as applicable to a relevant modelling phase.

NOTE The concept of specialization used in this document is less constrained than the specialization concept found in the domain of programming languages and object-oriented modelling.

5.12 Complementary concepts

The construct templates make use of several terms for conceptual representations that are not themselves constructs but are complementary and have a distinct significance and semantics for the purposes of enterprise modelling. These complementary concepts are as follows

— behavioural rule (see 7.2.5);

— constraint;

— declarative rule;

— integrity rule; and

— objective.

6 Conceptual structure

6.1 The four-component model

The classes underpinning this document are organized in terms of a meta-model and three sets of constructs – the core constructs, the derived constructs and the views. The Metamodel, Core and Derived components are normative sets (represented as packages in the illustrative UML class model of Figure 1) and the views are informative (represented as package View component).

NOTE There are many ways in which this conceptual structure can be represented, but this document uses UML class diagrams[9] as a notation that is well specified and widely accepted. Attributes and relationships are defined and normative within the normative parts of this document. They are illustrated using UML in informative figures and can be rendered in different ways in other modelling languages. UML is not necessarily intended as the basis of engineered instantiations, or as a starting point for the development of tools.

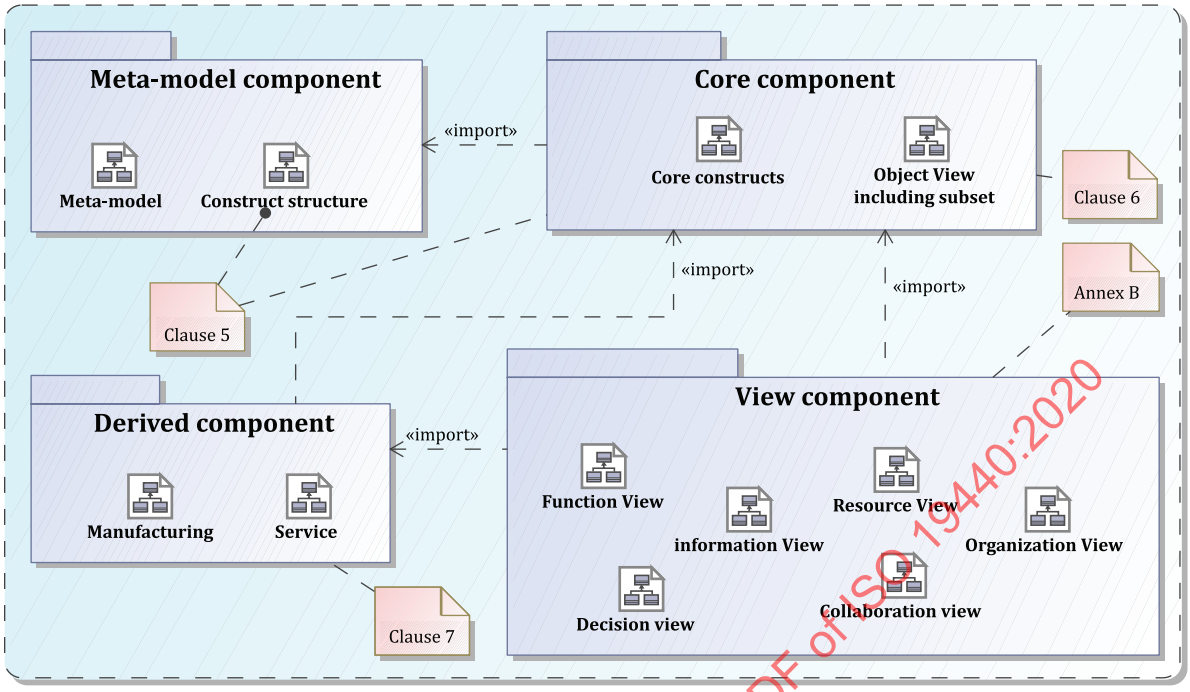
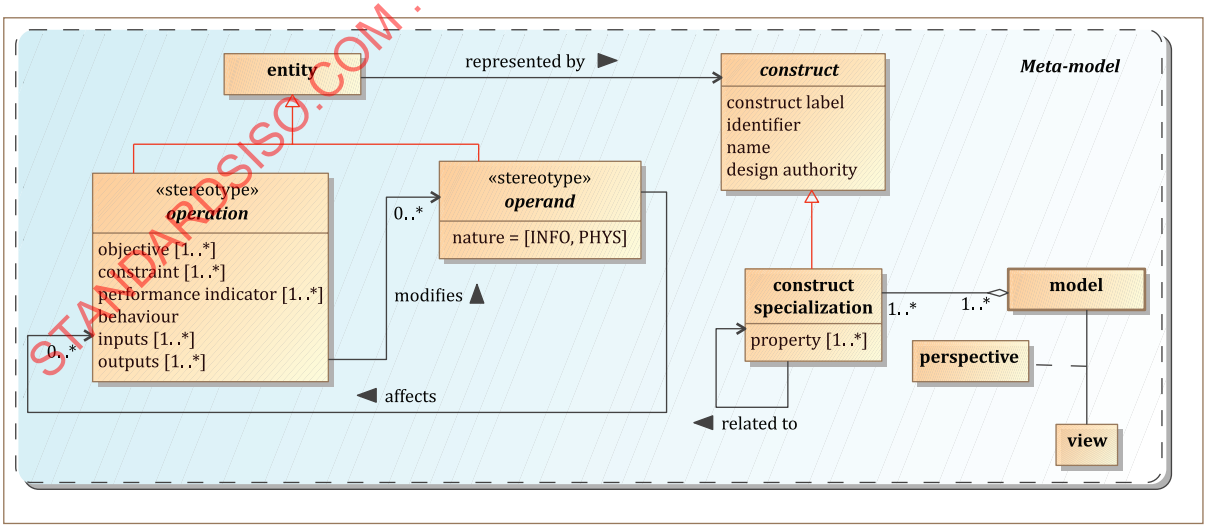


Figure 1 — Four component model

The <<import>> relations between the packages indicate that classes and relations in the meta-model are inherited by the core constructs and in turn those are both inherited by derived constructs and views.

6.2 Meta-model component

The normative components of the meta-model are illustrated as a UML diagram in Figure 2.



NOTE The inheritance relationships are coloured red to make them more visually distinguishable from other relationships.

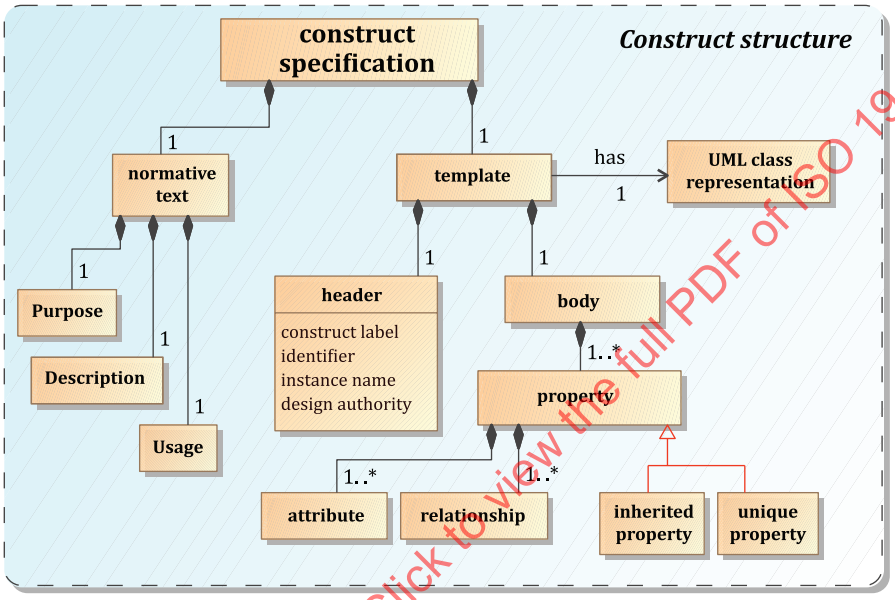
Figure 2 — Meta-model

Any entity in the enterprise domain is extended by being stereotyped as either an operation seeking to achieve certain objectives by modifying and in turn being affected by operands, or as an operand having properties that are changed by the behaviour of the operation.

NOTE In UML, "stereotypes" refer to how an existing meta-class can be extended so allowing an additional notation to be used.

An entity can be represented by a construct, which can be specialized. Collections of these constructs and specializations can be combined into an enterprise model. A model can be simplified by using a perspective to produce a View that represents selected portions of that model to reflect an orientation relative to an identified domain of interest. (Views are defined in ISO 15704.)

A construct has a structure that is illustrated in Figure 3.



NOTE The UML class representation is not normative.

Figure 3 — Construct structure

A construct is specified in terms of normative text and a template defining its properties and immediate relationships with other constructs. The template contains a header that is common to all constructs and a body containing the attribute and relationship properties. Properties can either be inherited from generalizations or stereotypes of that construct or be unique to that construct.

6.3 Core component

The set of fifteen core constructs is illustrated in Figure 4.

The meta-model class labelled as "construct" is specified as being an abstract element (denoted by italicization in Figure 4), meaning that only instances of construct are defined as core constructs in this document. A model user cannot create new core constructs, although existing constructs may be specialized and construct properties may be modified, over-ridden or extended to meet new modelling requirements. Each core construct is specified in Clause 7.

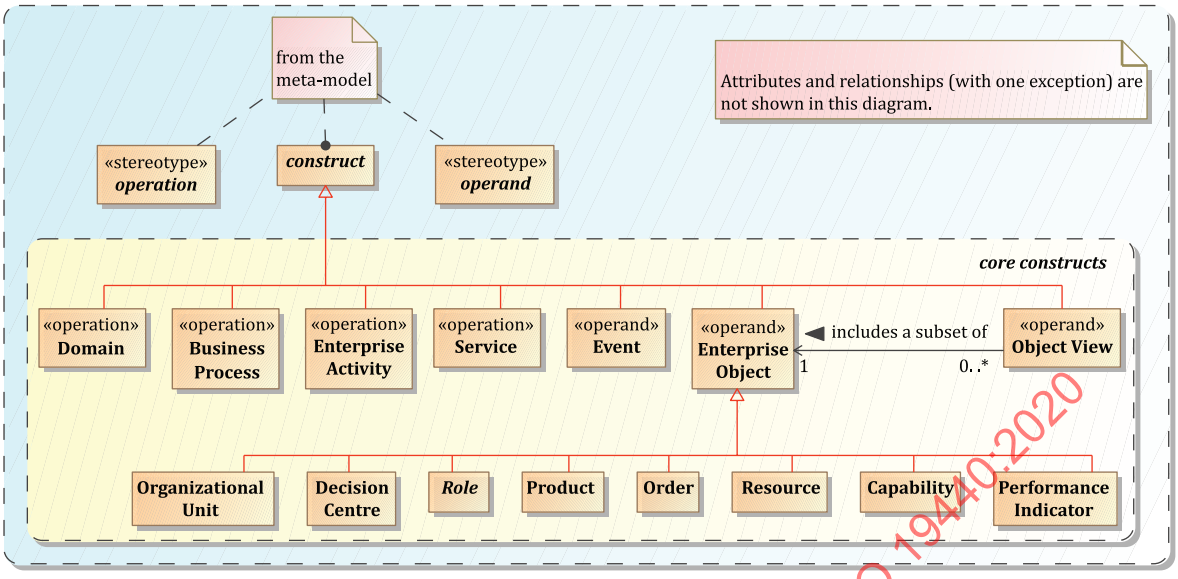


Figure 4 — Core constructs

Domain, Business Process and Service are stereotyped as operations having behaviour, inputs and outputs. Event, Enterprise Object and Object View are generated or modified by operations and so are stereotyped as operands. An Enterprise Object is specialized into Organizational Unit, Decision Centre, Role, Order, Product, Capability, Resource. and Performance Indicator.

NOTE Inherited properties are not shown explicitly in these UML diagrams.

Figure 4 omits construct attributes – those are shown in Figure 7. With one exception, Figure 4 also omits relationships between core constructs – those are shown in Figure B.1. The exception, the relationship between Enterprise Object and Object View, is because 'includes a subset of' is somewhat similar to a 'kind-of' inheritance relationship but not the same, since only some of the properties are inherited.

Since an Object View is a specialization of an Enterprise Object, a purpose-specific Object View is possible for any other specialization of an Enterprise Object as well, i.e. each specialization of an Enterprise Object may have specialized Object Views.

An Object View is created for a specific modelling purpose and suppresses the inherited attributes and relationships of the parent Enterprise Object that are not necessary for the intended purpose of the Object View. The Object View construct is illustrated in Figure 5.

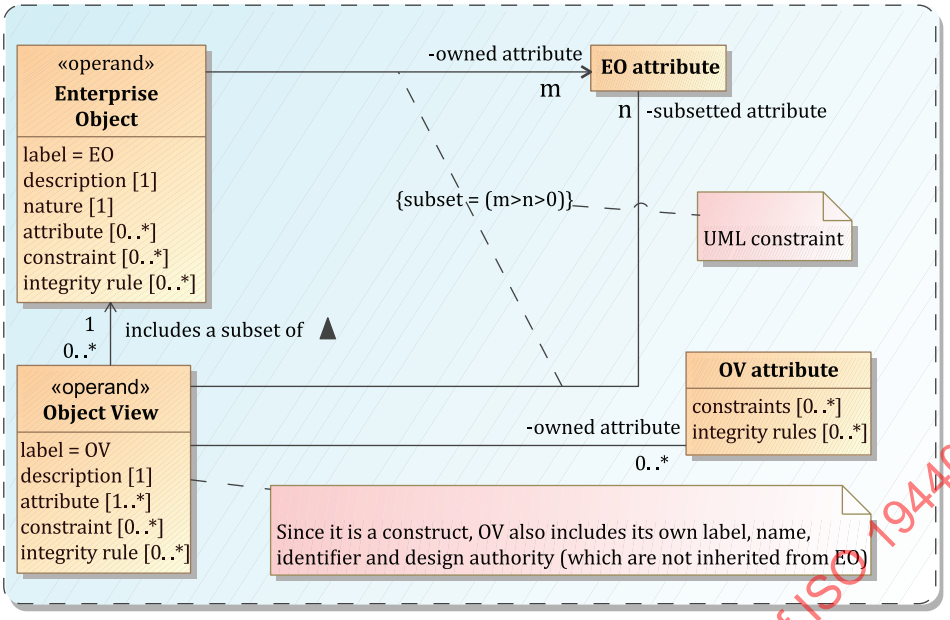


Figure 5 — Object View includes a subset of Enterprise Object

6.4 Derivations component

The set of core constructs can be further specialized as illustrated in Figure 6, and combined to form constructs, e.g. for application domains or distinct modelling purposes. In this document, such constructs are termed derived constructs and those for manufacturing- and service-oriented enterprises are specified in 8.2 and 8.3.

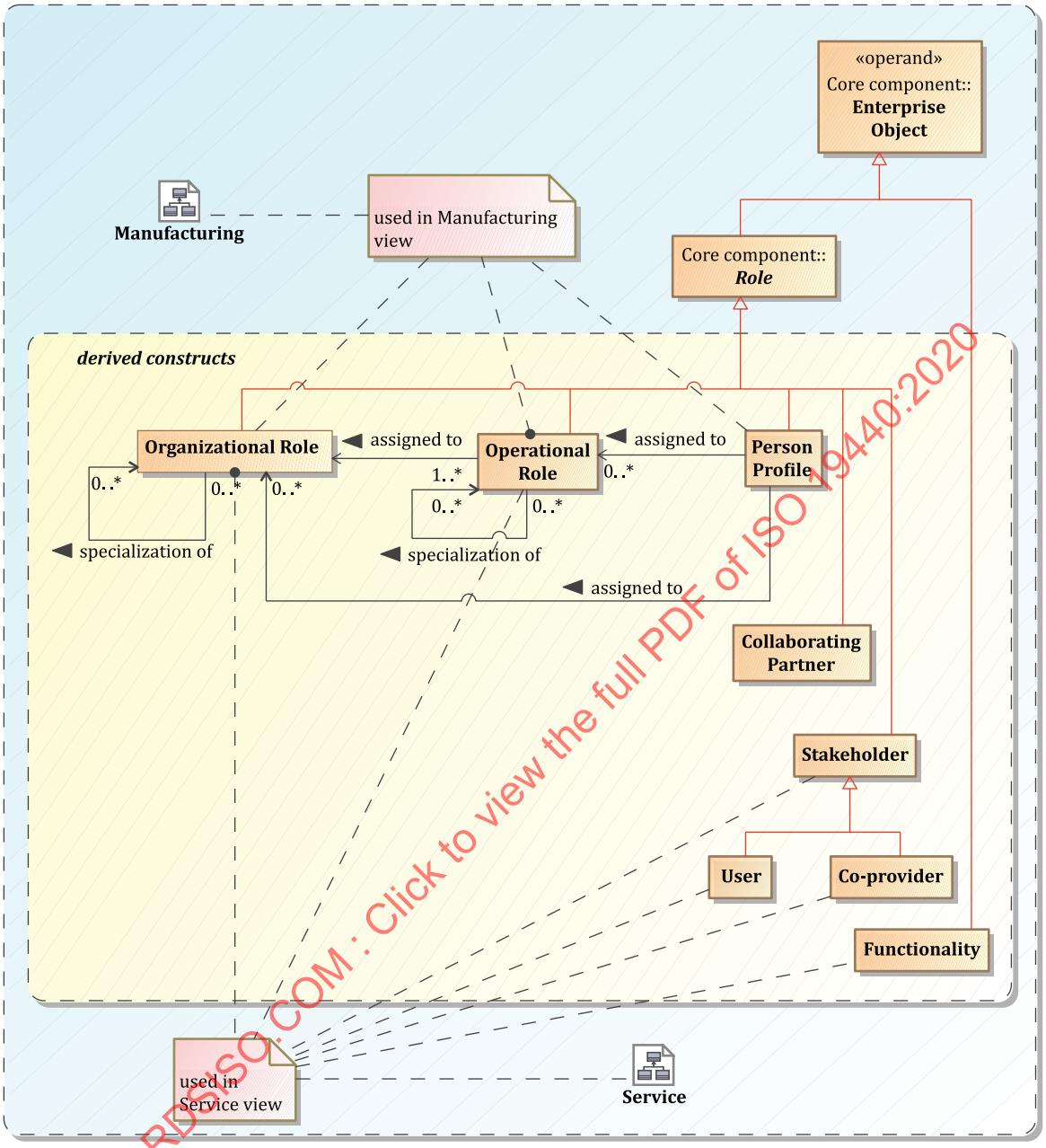


Figure 6 — Derived constructs

Figure 6 omits construct attributes – those are shown in Figure B.2.

6.5 Views component

While the Enterprise Object View construct supports creation of a construct containing a portion of the attributes and relationships of a particular Enterprise Object, the modelling concept of an enterprise model view does not create a new construct instance. Rather, an enterprise modelling view manifests selected content from several constructs, including both attributes and relationships, as a fragment of the source model from which they arise. If model content is compatible, an enterprise model view may span several source models.

The open-ended set of views described in Annex B contains a view-based analysis of a manufacturing-oriented enterprise and example decision and collaboration views which also contain further constructs.

7 Core constructs

7.1 Domain

7.1.1 Purpose

The purpose of the domain construct is to represent the portion of an enterprise to be modelled, providing for identification of the relevant information. The information describes the boundary and the content of an enterprise or a portion of an enterprise for which the model will be created.

7.1.2 Description

A Domain construct shall describe the parts of the enterprise to model and the relationships with the external environment from a high-level management-oriented perspective. The model is subject to the business model of the enterprise, usually expressed by a given set of strategic and operational business objectives and constraints, and the purpose for which the model is created. The relations between strategic and operational objectives should be made visible as much as possible. The Domain inputs and outputs and their origins and destinations define the domain boundaries and the relations between inputs and outputs identify the required functionalities – the Business Processes and Services of the domain.

7.1.3 Usage

All model life cycle phases use the Domain construct. The Domain template shall enable

a) the capturing of all inter- and intra-organizational-related information relevant for strategic planning and decision support, and

b) its presentation in a form that is understandable by both humans and interpretable by digital technology.

The extent of attribute and relationship specification detail should be appropriate for the enterprise model phase applicable to the Domain construct's usage. The inclusion of an Event or Object View implies that the Event or associated Enterprise Object is also defined at the same model phase and level of detail.

7.1.4 Construct template for Domain

Table 1 — Domain template

Header	
Construct label	<Model-unique abbreviation> DM
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. DM.1-2.
Name	[<adjective>]? <noun> Name of Domain, where <noun> indicates the scope of the Domain, and <adjective> optionally qualifies the Domain.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having authority to design or maintain this Domain.

Table 1 *(continued)*

Body	
Attributes relevant for domain identification and later phases	
Description	<string> Short textual description of the Domain.
Domain Inputs	[<name> [<short textual description>]?]+ Name and optionally short description of the main Domain inputs.
Domain Outputs	[<name> [<short textual description>]?]+ Name and optionally short description of the main Domain outputs.
Main Processes	[[<identifier>'/'<name>] [<functionality text>]?]+ List of Business Process and Services, each followed by an optional description of its functionality. The functionality text is a short description of the main functionality needed to transform Domain inputs into Domain outputs. The list of identified Business Processes or Services is restricted to those that are required to achieve the Domain objectives.
Objectives	[<objective>]+ Strategic and operational business objectives of this Domain.
Constraints	[<constraint>]* Constraints imposed on this Domain.
Attributes applicable at concept definition and later phases	
Domain Business Model	['Mission: ' <mission text>] ';' ['Vision: ' <vision text>] ';' ['Values: ' [<value text>]+] Short textual description of the domain's mission, vision and values.
Domain Operation	['Strategies: ' <strategy text]* ';' ['Policies: ' <policy text>]* ';' ['Operational Concepts: ' [<operational concept]* ';' ['Business Plans: ' [<business plan]* Short textual description of the various strategies, policies, etc.
Decision Function Category	[[<verb>]? <noun>]* Name of the decision function categories that belong to the Domain.
Relationships relevant for domain identification and later phases	
Inputs/Outputs	
Object View Inputs	[<origin> ':' <identifier> '/' <name>]+ Identifiers and names of origins and Object Views which are available for this Domain at the boundary to the external environment.
Event Inputs	[<origin>] ':' <identifier> '/' <name>]+ Events, which can be received by this Domain from the external environment.
Object View Outputs	[<destination> ':' <identifier> '/' <name>]+ Object Views, which are made available by this Domain at the boundary to the external environment.
Event Outputs	[<destination> ':' <identifier> '/' <name>]+ Events which can be generated by this Domain for the external environment.
Relationships applicable at requirements definition and later phases	
Uses Business Processes	[<identifier> '/' <name>]* Business Processes used in this Domain.
Uses Services	[<identifier> '/' <name>]* Services used in this Domain.
Uses Performance Indicators	[<identifier> '/' <name>]+ Performance Indicators which are available for this Domain. *Performance indicators identify the required value or method by which achievement of the objectives can be assessed.*
Relationships applicable at design specification and later phases	
Operational Relationships	

Table 1 *(continued)*

Operation Responsibility	<identifier> '/' <name>] ':' [<identifier> '/' <name>] Operational Role and Organizational Unit respectively, having responsibility for operation of this Domain.
Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Role and Organizational Unit respectively, having authority for this Domain.

NOTE 1 For Objectives, derived objectives occur in Business Processes, Enterprise Activities, Services and Decision Centres as well; consistency needs to be assured.

NOTE 2 In this and the following templates, <origin> in Relationships denotes the Domain, Business Process, Enterprise Activity or Service that is the source of the input. In the situation that the <origin> is an external Domain, only the name of the Domain is stated, and its contents are not further defined. Similarly, <destination> denotes the Domain, Business Process, Enterprise Activity or Service that will receive the output.

NOTE 3 A Domain specifies at least one Business Process or Service. It is the responsibility of the tool supporting model development and maintenance to ensure that the Business Processes and Services listed here are compatible with the Used by attribute of Business Processes and Services.

7.2 Business Process

7.2.1 Purpose

The purpose of the Business Process construct is to represent a partially ordered set of Business Processes, Services and Enterprise Activities that execute to realize one or more given objectives of an enterprise or a part of an enterprise to achieve some desired end-result. The highest-level Business Processes and Services in a domain realize all or part of the domain functionalities, together with its internal structure and its dynamic behaviour.

7.2.2 Description

A Business Process construct shall describe the functionalities needed to produce a desired result that satisfies one or more business objectives derived from business objectives defined for the enterprise domain. This result emerges from transformations or combinations, or both, of entities into new entities or into new states that require appropriate control and functional capacity.

The construct shall identify all used and constituent Business Processes, Services and Enterprise Activities that represent the decomposition of the Business Process functionality according to modelling criteria and the needs of operational monitoring and control into a number of ordered transformation actions, thus capturing relevant intermediate conditions of the entities to be changed.

These transformation actions shall be characterized as combinations of used and constituent Business Processes, Services or Enterprise Activities and their interconnections, arranged by ordering relationships and dependencies described by behavioural rules (see 7.2.5), which capture the process dynamics. The construct shall describe in its behavioural rule set attribute, the logical binding of its Business Processes, Services or Enterprise Activities to the behavioural rules connecting these intermediate conditions, i.e. the logical sequence of these transformation functions.

NOTE Clause A.1 specifies the rationale for restrictions and constraints imposed on behavioural rules.

The result of a Business Process shall be observable or quantifiable. It can result in material entities (such as products) and information entities (such as orders, documents or data), both being represented as Enterprise Objects or specializations thereof (e.g. Product).

A Business Process may have an ending status, but this is exceptional. Process completion is usually signalled by a behavioural rule FINISH or GENERATE <exception> AND FINISH.

7.2.3 Usage

The construct Business Process is used in all model phases. Its template shall enable

a) the capturing of all process-related information relevant for model-based planning and decision support, and monitoring and control of operational processes, and

b) its presentation in a form that is understandable by humans and interpretable by digital technology.

7.2.4 Construct template for Business Process

Table 2 — Business Process template

Header	
Construct label	<Model-unique abbreviation> BP
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. BP_2
Name	[<adjective>]? <noun> Name of Business Process, where <noun> indicates the scope of the Business Process, and <adjective> optionally qualifies the Business Process.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having the authority to design or maintain this Business Process.
Body	
Attributes relevant for domain identification and later phases	
Description	<string> Textual description of functionalities and desired result.
Objectives	[<objective>]+ Strategic and operational business objectives to be fulfilled by the Business Process.
Attributes applicable at requirements definition and later phases	
Constraints	[<constraint>]* Constraint imposed on the Business Process.
Declarative Rule	[<declarative rule>]* Declarative rules applicable to this Business Process.
Behavioural Rule Set	[<behavioural rule>]+ Collection of behavioural rules expressed using the syntax defined in 7.2.5 and Annex A. (*The collection of behavioural rules constitutes the behavioural rule set.)*
Activation priority	<integer> Integer in a system-defined range representing the lowest and highest priorities respectively.
Ending Status	[<value> [',' <priority>]?]* Possible ending status values produced by this Business Process, where <value> is a mandatory 0-argument predicate and <priority> is an integer in a system-defined range representing the lowest and highest priorities respectively. By default, highest priority.
Relationships relevant for domain identification and later phases	
Domain usage	[<identifier> '/' <name>]* Domain(s) using this Business Process.
Relationships at requirements definition and later phases	
Uses Performance Indicators	[<identifier> '/' <name>]* Performance Indicators by which achievement of the objectives can be assessed.

Table 2 *(continued)*

Part-of	[<identifier> '/' <name>]* Business Process in which this Business Process is involved in an aggregation. The name of the Business Process being defined cannot be used in the list.
Uses Services	[<identifier> '/' <name>]* Services used by this Business Process.
Consists-of	[<identifier> '/' <name>]* Business Processes, Enterprise Activities and Services of which this Business Process is the aggregate.
Inputs/Outputs	
Object View Inputs	[<origin>] ':' <identifier> '/' <name>]+ Object Views which are available for this Business Process.
Event Inputs	[<origin>':' <identifier> '/' <name>]+ Events which can be received by this Business Process.
Object View Outputs	[<identifier> '/' <name> ':'<destination>]+ Object Views which are made available by this Business Process.
Event Outputs	[<identifier> '/' <name> ':'<destination>]+ Events which can be generated by this Business Process.
Relationships applicable at design specification and later phases	
Operational Relationships	
Operation Responsibility	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Operational Role and Organizational Unit respectively, having responsibility for operation of this Business Process.
Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Role and Organizational Unit respectively, having authority for this Business Process.

NOTE 1 Activation priority attributes are used at run-time to select which Business Process or specialization to activate first in the case of more than one Business Process being initiated by a Behavioural Rule set. They are also used to guide allocation of Resources to Enterprise Activities.

NOTE 2 Ending status priority is used at run-time to select which Business Process or specialization thereof to respond to first when two processes complete within the same interval of observable time.

NOTE 3 For Event Inputs and Outputs, it is the responsibility of the tool supporting model development and maintenance to ensure that the Events listed as inputs or outputs are compatible with the Generated-by attributes of Event.

7.2.5 Behavioural rules

7.2.5.1 Purpose

The purpose of behavioural rules is to identify the start of the main Business Process and to describe the logical sequencing relationships of constituent Business Processes, Services and Enterprise Activities used in the specification of Business Process behaviour.

7.2.5.2 Description

The behaviour of a Business Process shall be described in its behavioural rule set attribute by a set of behavioural rules that control the sequencing of constituent Business Processes and Enterprise Activities.

The logical sequencing may be more or less well specified, corresponding to:

— well-structured processes, that is, processes for which the expected result is known and the sequence of constituent and used Business Processes, Services and Enterprise Activities is completely defined;

— semi-structured processes, that is, processes for which the expected result is known and the sequence of constituent and used Business Processes, Services and Enterprise Activities will be known only at run-time; and

— ill-structured processes, that is, processes for which neither the result nor the sequence of constituent and used Business Processes, Services and Enterprise Activities is completely known.

Annex A defines constraints and provides a detailed description and syntax for these rules.

7.2.5.3 Usage

The concept of behavioural rules is used in the requirements definition phase and later phases. The rules shall enable

a) the capturing of all the conditions that control the sequencing of Business Processes, Services and Enterprise Activities,

b) the capturing of the consequent dynamic behaviour of Business Processes, Services and Enterprise Activities and

c) the presentation of conditions and behaviour in a form that is understandable by humans and interpretable by digital technology.

7.3 Enterprise Activity

7.3.1 Purpose

The purpose of the Enterprise Activity construct is to represent the part of process functionality that is needed to realize a basic task within a Business Process or Service of an enterprise domain. In contrast to Business Processes (see 7.2), that are concerned with enterprise objectives, Enterprise Activities are concerned with production requirements and are defined according to required user objectives for operational monitoring and control.

7.3.2 Description

An Enterprise Activity construct shall describe all things required for and produced by the execution of a specific task, which transforms function input(s) into function output(s) using control, operational role and resource inputs and optionally producing control, operational role and resource-related information outputs. These outputs shall contain status information on the execution of the task as they relate to the activity itself and the resource.

The internal task behaviour of an Enterprise Activity is described in the activity behaviour attribute.

If an Enterprise Activity is decomposed into other Enterprise Activities, it shall identify any constituent Enterprise Activities and characterize their interactions by ordering relationships and dependencies described by behavioural rules, which capture the process dynamics.

Operational Role inputs shall identify required skills to be provided by assigned Person Profiles.

An Enterprise Activity shall have an ending status.

7.3.3 Usage

The construct Enterprise Activity is used in the requirements definition phase and later phases. Its template shall enable the capturing of all relevant information inputs and outputs needed and created in the execution of the identified functionality of this Enterprise Activity, and the representation of relationships with model-based, and monitoring and control of operational processes.

Such information should be presented in a form that is understandable by humans and processable by computer. In addition, where the functionality needs to be decomposed, the template shall identify all

sub-Enterprise Activities that represent the decomposition of the Enterprise Activity functionality and are needed for Resource and Operational Role assignments.

7.3.4 Construct template for Enterprise Activity

Table 3 — Enterprise Activity template

Header	
Construct label	<Model-unique abbreviation> EA
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. EA_3
Name	<verb> [[<adjective>]? <noun>]? Name of this Enterprise Activity, where <verb> characterizes the imperative nature of the Enterprise Activity, <adjective> and <noun> are optional qualifier and scope indications respectively of the Enterprise Activity.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having authority to design or maintain this Enterprise Activity.
Body	
Attributes applicable at requirements definition and later phases	
Description	<string> Textual description.
Activity Behaviour	<description> | <specification of tasks> Description of the task to be undertaken or specification of the sequencing of constituent sub-activities (tasks).
Objectives	[<objective>]+ Operational business objectives to be fulfilled by the Enterprise Activity.
Constraints	[<constraint>]* Constraints imposed on the Enterprise Activity.
Attributes applicable at design specification and later phases	
Behavioural Rule Set	[<behavioural rule>]+ Behavioural rule(s), expressed using the syntax as defined in 7.2.5 and Annex A. *(The collection of behavioural rules constitutes the behavioural rule set.)*
Activation priority	<integer> Integer in a system-defined range representing the lowest and highest priorities respectively.
Ending Statuses	[<value> [',' <priority>]?]* Possible ending status values produced by this Enterprise Activity, where <value> is a mandatory 0-argument predicate and <priority> is an optional integer in a system-defined range representing the lowest and highest priorities respectively. By default, highest priority.
Duration	[<duration> <qual>]* Attribute pairs defining the duration of this Enterprise Activity, where <qual> is a code representing one of 'average', 'minimum', 'maximum', 'actual'.
Relationships applicable at requirements definition and later phases	
Uses Performance Indicators	[<identifier> '/' <name>]+ Performance Indicators by which achievement of the objectives can be assessed.
Used by	[<identifier> '/' <name>]+ Business Processes, Services and Enterprise Activities using this Enterprise Activity.

Table 3 *(continued)*

Inputs	
Information Inputs	[<origin>':' <identifier> '/' <name>]+ Object Views describing input information to be processed by this Enterprise Activity.
Control Inputs	[<origin>':' <identifier> '/' <name>]+ Object Views describing input information providing run-time data used but not modified by occurrences of this Enterprise Activity, for example Order requirements constraining and controlling this Enterprise Activity.
Required Capabilities	[<identifier> '/' <name>]+ Capability or Capabilities defining required capabilities of occurrences of this Enterprise Activity.
Resource Inputs	[<origin>':' <identifier> '/' <name>]+ Resources required by occurrences of this Enterprise Activity. *Resource inputs identify the provided capabilities (see 7.13 and 7.14).*
Input Events	[<origin>':' <identifier> '/' <name>]+ Events which can be received by occurrences of this Enterprise Activity.
Relationships applicable at design specification and later phases	
Consists-of	[<Enterprise Activity> [':' [<param>]+]?]* Enterprise Activities which this Enterprise Activity is the aggregate.
Outputs	
Information Outputs	[<identifier> '/' <name> ':'<destination>]+ Object Views describing the changes to Enterprise Objects produced by occurrences of this Enterprise Activity. *The information outputs contain status information on the execution of the task and subtasks as they relate to the activity and associated resources.*
Control Outputs	[<identifier> '/' <name> ':'<destination>]+ Object Views describing the status information of the Enterprise Activities after the execution of this Enterprise Activity.
Resource Outputs	[<identifier> '/' <name> ':'<destination>]+ Object View describing the status information of Resources after the execution of this Enterprise Activity.
Output Events	[<identifier> '/' <name> ':'<destination>]+ Events which can be generated by this Enterprise Activity.
Operational Relationships	
Operation Responsibility	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Operational Role and Organizational Unit respectively, having responsibility for operation of this Enterprise Activity.
Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Role and Organizational Unit respectively, having authority for operation of this Enterprise Activity.

NOTE 1 The control flow of these tasks collectively called activity behaviour can be defined in an algorithm, which describes the input/output conditions for sequence control of sub-tasks involved in the execution of the Enterprise Activity. The specification of the algorithm is considered to be an implementation issue beyond the scope of this document.

NOTE 2 Activation priority attributes are used at run-time to select which process to activate first in the case of more than one process being initiated by a Behavioural Rule set. They are also used to guide allocation of Resources to Enterprise Activities.

NOTE 3 For Resource Inputs, it is the responsibility of the designer with supporting tools to ensure that these Resources include at least those specified as able to provide the required capabilities.

7.4 Service

7.4.1 Purpose

The purpose of the Service construct is to represent a functionality that can be bought (or sold) and is the result of a process that includes at least one action that is carried out at the interface between the supplier (provider) and the user (consumer). The service template includes service information collected from stakeholders and end-users. Benefits for the user are considerable quality improvement in the design process and cost reduction in the system operation resulting from a coordinated use of a common set of constructs.

NOTE More detailed and specific information than that contained in the template can be modelled using relevant modelling languages or tools such as Blueprint, UML, use case diagrams, etc.

7.4.2 Description

A Service construct shall describe the service system providing functions that fulfil the needs of a User (see 8.4.5) with respect to a Product (see 7.11).

In a service system,

— various kinds of Business Processes, other Services and Enterprise Activities are necessary to provide the functionality of the service, such as service booking process, service delivery process, service planning process, supply process, or maintenance process, and

— various entities are involved in such a development, ranging from the User who will consume the service to the provider and co-providers, and to other entities such as technical centres, research centres and banks; all these entities are called stakeholders, and each may express specific concerns.

A Service constructs includes all things required for and produced by the execution of a specific task, which transforms function input(s) into function output(s), using control, operational role (a specialization of Role) and resource inputs and optionally producing control, operational role and resource related information outputs. The latter shall contain status information on the execution of the task as they relate to the activity itself and the resource, respectively. For a manufacturing-oriented enterprise (see 8.2), Operational Role inputs identify required skills to be provided by assigned Person Profiles. Resource inputs and outputs shall identify the required and provided capabilities, respectively.

The Service construct shall also identify all constituent Business Processes, Services and Enterprise Activities that represent the decomposition of the Service functionality according to modelling criteria and the needs of operational monitoring and control into a number of ordered transformation actions, thereby capturing relevant intermediate conditions of the entities to be changed.

These transformation actions shall be characterized as combinations of constituent and used Business Processes, Services or Enterprise Activities and their interconnections, arranged by ordering relationships and dependencies described by behavioural rules, which capture the process dynamics. The construct shall describe in its behavioural rule set attribute, the logical binding of its constituent and used Business Processes, Services or Enterprise Activities to the behavioural rules connecting these intermediate conditions, i.e. the logical sequence of these transformation functions.

7.4.3 Usage

The Service construct is used in all model phases. Its template shall enable

a) the capturing of all service-related information relevant for the processes and entities that are involved in a service system, and

b) its presentation in a form that is understandable by humans and interpretable by digital technology.

7.4.4 Construct template for Service

Table 4 — Service template

Header	
Construct label	<Model-unique abbreviation> SRV
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. SRV_3
Name	<verb> [[<adjective>]? <noun>]?] Name of this Service, where <verb> characterizes the imperative nature of the Service, <adjective> and <noun> are optional qualifier and scope indications respectively of the Service.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively having authority to design or maintain this Service.
Body	
Attributes relevant for domain identification and later phases	
Description	<string> Short textual description of the Service.
Objectives	[<objective>]+ Operational business objectives to be fulfilled by the Service.
Attributes relevant for different enterprise model phases	
Attributes applicable at concept definition and later phases	
Constraints	[<constraint>]* Constraint(s) imposed on the Service.
Attributes applicable at design specification and later phases	
Behavioural Rule Set	[<behavioural rule>]+ Behavioural rule(s), expressed using the syntax as defined in 7.2.5 and Annex A. *(The collection of behavioural rules constitutes the behavioural rule set.)*
Attributes applicable at requirements definition and later phases	
Declarative Rules	[<rule>]* Declarative rules applicable to this Service
Functionality	[<functionality text>]+ | [Functionality derived construct] Text describing the functionality or [defined only in a Servitization specialization (8.3)] reference to Functionality derived construct (see 8.4.7).
Value	[<text>]+ | [Performance Indicator]+ Text describing the value of the service or reference(s) to Performance Indicator(s) (see 7.15).
Attributes applicable at design specification and later phases	
Service Attributes	[<attribute_name> [' = ' [<attribute_value>]+]?] Elements representing service attributes and possibly their values for the entity represented by the Service.
Ending Status	"[<value> [',' <priority>]?]* Ending status values produced by occurrences of this Service, where <value> is a mandatory 0-argument predicate and <priority> is an optional integer in a system-defined range representing the lowest and highest priorities respectively. By default, highest priority.
Relationships relevant for domain identification and later phases	
Uses Performance Indicators	[<identifier> '/' <name>]+ Performance Indicator by which achievement of the objectives can be assessed.

Table 4 *(continued)*

Used by	[<identifier> '/' <name>]* Service(s) and/or Business Process(es) using this Service.
Relationships applicable at requirements definition and later phases	
Products	[<identifier> '/' <name>]* Products with which this Service is concerned, where multiplicity [...]* is one of [0..*] (only for early modelling phases) or [1..1] or [1..*] or [m..n].
User	[<identifier> '/' <name>]* User (s) (see 8.4.5) using the service.
Stakeholder	[<identifier> '/' <name>]* Stakeholder (s) (see 8.4.4) concerned with this Service.
Part-of	[<identifier> '/' <name>]* Services in which this Service is involved in an aggregation. *The name of the Service being defined cannot be used in the list.*
Consists-of	[<identifier> '/' <name>]* Services, Business Processes and Enterprise Activities of which this Service is the aggregate.
Inputs/Outputs	
Information Inputs	[<origin>] ':' [<identifier> '/' <name>]* Object Views describing: a) information to be processed by this Service, b) control information providing run-time data used but not modified by occurrences of this Service, c) Resources used or consumed by this Service.
Event Inputs	[<origin>] ':' <identifier> '/' <name>]* Events which can be received by this Service.
Information Outputs	[<identifier> '/' <name> ':'<destination>]+ Object Views describing:. a) the information outputs produced by occurrences of this Service (these contain status information on the execution of the operational components as they relate to the process and associated resources), or b) status information of Resources after the execution of this Service.
Event Outputs	[<identifier> '/' <name> ':'<destination>]+ Events which can be generated by this Service.
Relationships applicable at design specification and later phases	
Operational Relationships	
Operation Responsibility	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Operational Role and Organizational Unit respectively, having responsibility for operation of this Service.
Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Role and Organizational Unit respectively, having authority for operation of this Service.

NOTE 1 Clause A.1 specifies the rationale for and constraints imposed on behavioural rules.

NOTE 2 User and Stakeholder are defined only in the Servitization specialization (see 8.3).

7.5 Event

7.5.1 Purpose

The purpose of the Event construct is to represent a solicited or unsolicited fact indicating a state change in the enterprise or its environment. Its purpose is to capture the origin and destination of an event.

7.5.2 Description

An Event construct shall describe the initiation of a state change in the enterprise or its environment, for example to initiate the execution of one or more processes, and shall activate Business Processes, Services or Enterprise Activities by initiating the processing of the behavioural rule set associated with a Business Process, Service or Enterprise Activity. Object Views shall be used to convey any information that is associated with an Event.

An Event is also used to signal an internal or external occurrence of significance to a Domain.

7.5.3 Usage

The construct Event is used in all model phases. Its template shall capture the event-relevant information and present it in a form that is understandable by humans and interpretable by digital technology.

NOTE 1 Some Events can be simply signals that something has occurred while others have associated information that will be conveyed by an Object View, e.g. representing an Order.

EXAMPLE Event: arrival of customer order (Object View: Customer order information).

NOTE 2 The Event construct does not include information regarding its duration, persistence, timeout, or other attributes concerning its utilization, which are all implementation dependent features.

Every Event is a unique occurrence. Events shall be given a name, which may be qualified by an adjective, to indicate some specific structure and behaviour.

7.5.4 Construct template for Event

Table 5 — Event template

Header	
Construct label	<Model-unique abbreviation> EV
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. EV_4
Name	<adjective>]? <noun> Name of the Event, where <noun> indicates the entity causing the Event, and <adjective> is an optional qualifying attribute.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having authority to design or maintain this Event.
Body	
Attributes relevant for domain identification and later phases	
Description	<string> Short textual description of the Event.
Attributes applicable at design specification and later phases	

Table 5 *(continued)*

Timestamp	<YYYY-MM-DDThh:mm:ss.sss> Data type defining date and time to be provided at occurrence creation time, where YYYY represents calendar years, MM represents month, DD represents days, "T" separates date from time, hh represents hours (using a 24 hour clock), mm represents minutes and ss.sss represent seconds to three decimal places, as in 2018-11-25T11:44:05.123. *Timestamp has a fixed value, which is a data type for date and time (and cannot be changed because it is known by the system).*
Priority	<integer> Integer in a system-defined range representing the lowest and highest priorities respectively. By default, highest priority.
Relationships relevant for domain identification and later phases	
Object Views	[<origin> ':' <identifier> '/' <name>]* Object View (s) defining information that can be associated with occurrences of this Event.
Generated_by	[<origin>':' <identifier> '/' <name>]+ Source (s) of this Event.
Initiates	<identifier> '/' <name> ':'<destination> Destination that can receive this Event.
Relationships applicable at design specification and later phases	
Operational Relationships	
Operation Responsibility	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Operational Role and Organizational Unit respectively, having responsibility for operation of this Event.
Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Role and Organizational Unit respectively, having authority for this Event.

NOTE The Timestamp is formatted using ISO 8601.

7.6 Enterprise Object

7.6.1 Purpose

The purpose of the Enterprise Object construct is to describe objects in the enterprise through their characteristics (attributes and relationships) and to provide for selection of relevant parts [Object Views (see 7.7)] that need to be identified in the modelling process and used during the operational phase.

7.6.2 Description

An Enterprise Object construct shall describe the common characteristics of a thing, an enterprise entity, as it exists during its lifetime. Its usage shall be restricted to those situations where only the information aspects of the entity under consideration are relevant.

7.6.3 Usage

The construct Enterprise Object is used in all model phases. Its template shall enable

a) the capturing of all relevant information and relations that are needed for model-based planning and decision support, monitoring and control of operational processes and their execution, and

b) its presentation in a form that is understandable by humans and interpretable by digital technology.

The attribute 'Nature of Object' shall be used to express an Enterprise Object's primary characteristic that determines how easily the entity that it represents can be transported, replicated, etc. The term

is used to differentiate the information flow from the physical flow among Business Processes or Enterprise Activities.

7.6.4 Construct template for Enterprise Object

Table 6 — Enterprise Object template

Header	
Construct label	<Model-unique abbreviation> EO
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. EO.5
Name	<noun> Name of the Enterprise Object.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having authority to design or maintain this Enterprise Object.
Body	
Attributes relevant for domain identification and later phases	
Description	<string> Short textual description.
Nature of Object	[PHYSICAL | INFORMATION] Nature of the entity represented by the Enterprise Object.
Attributes	[<attribute_name> [' = ' [<attribute_value>]+]?]* Elements representing attributes and possibly their values for the entity represented by the Enterprise Object.
Constraints	[<constraint>]* Constraint (s) imposed on selected named attributes of the Enterprise Object.
Attributes applicable at requirements definition and later phases	
Integrity Rules	[<integrity rule>]* Integrity rule (s) applicable to attributes of the Enterprise Object in the requirements definition phase.
Relationships relevant for domain identification and later phases	
Class Relationships – list of Enterprise Object relationships in the form:	
Specialization-of	[<identifier> '/' <name>]* Enterprise Objects that are generalizations of this Enterprise Object. *The name of the Enterprise Object being defined cannot be used in the list.*
Generalization-of	[<identifier> '/' <name>]* Enterprise Objects that are specializations of this Enterprise Object. *The name of the Enterprise Object being defined cannot be used in the list.*
Part-of	<identifier> '/' <name>]* Enterprise Object (s) for which this Enterprise Object is involved in an aggregation. *The name of the Enterprise Object being defined cannot be used in the list.*
Consists-of	[<identifier> '/' <name>]* Enterprise Objects of which this Enterprise Object is the aggregate. *The name of the Enterprise Object being defined cannot be used in the list.*
Related-to	[<identifier> '/' <name>]* Other construct instances that are related to this Enterprise Object, where multiplicity [...]* is one of [0..*] (only for early modelling phases) or [1..1] or [1..*] or [m..n].
Relationships applicable at design specification and later phases	
Operational Relationships	

Table 6 *(continued)*

Operation Responsibility	<identifier> '/' <name>] ':' [<identifier> '/' <name>] Operational Role and Organizational Unit respectively, having responsibility for operation of this Enterprise Object.
Authority	[<identifier> '/' <name>]* [<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Role and Organizational Unit respectively, having authority for this Enterprise Object.

7.7 Enterprise Object View (Object View)

7.7.1 Purpose

The purpose of an Object View construct is to enable the identification of the relevant subset of the properties of an Enterprise Object as required by a Domain, a Business Process, an Enterprise Activity or a Service for the definition of its inputs and outputs.

7.7.2 Description

An Enterprise Object View construct shall describe a partial subset of an Enterprise Object or specializations thereof by specifying selected attributes and relationships of that Enterprise Object or specialization, which may include constraints and integrity rules. Object Views shall also be used to convey any information that may be associated with an Event.

7.7.3 Usage

The construct Object View is used in all model phases. Its template shall enable

a) the capturing of relevant information needed for strategic planning and decision support, monitoring and control of operational processes, and

b) its presentation in a form that is understandable by humans and interpretable by digital technology.

NOTE The term 'Enterprise Object View' is usually shortened to 'Object View'.

7.7.4 Construct template for Object View

Table 7 — Object View template

Header	
Construct label	<Model-unique abbreviation> OV
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. OV_6
Name	<noun> Name of the Object View.
Design Authority	[<identifier> '/' <name>] [':' [< identifier> '/' <name>]]? Organizational Unit and possibly Organizational Role, having authority to design or maintain this Object View.
Body	
Attributes relevant for domain identification and later phases	
Description	<string> Short textual description.

Table 7 *(continued)*

Attributes	<attribute_name> ['= '<attribute_value>] ? Elements representing attributes of the Object View and optionally their values. *(These attributes are a subset of the attributes of the Enterprise Object to which the Object View applies).*
Constraints	[<constraint>]* Constraint (s) imposed on selected attributes of the Enterprise Object from which the Object View is derived.
Attributes applicable at requirements definition and later phases	
Integrity Rules	[integrity rule]* Integrity rule (s) applicable to the selected attributes of the Enterprise Object from which the Object View is derived.
Relationships relevant for domain identification and later phases	
Enterprise Object	[<identifier> '/' <name>] Enterprise Object of whose attributes the Object View includes a partial sub-set.
Relationships applicable at design specification and later phases	
Operational Relationships	
Operation Responsibility	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Operational Role and Organizational Unit respectively, having responsibility for operation of this Object View.
Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Role and Organizational Unit respectively, having authority for this Object View.

7.8 Organizational Unit

7.8.1 Purpose

The purpose of the Organizational Unit construct is to describe an identifiable entity of the enterprise organization together with its position relative to other such entities in the enterprise organizational structure.

7.8.2 Description

An Organizational Unit construct shall describe the formal, positional or administrative structure of an enterprise, or some combination thereof. Its content shall be determined by naming the assigned entities for which it has authority or responsibility, or both. Its position in the organization shall be represented by the definition of the assignments of lower level units to the current unit and its own assignment to higher-level units. The construct enables multidimensional organization structures (i.e. matrix organizations, network organizations and others) by allowing multiple relationships to other organizational units.

Each Organizational Unit shall contain at least one relationship to an Organizational Role specifying the required or provided organizational skills, responsibilities and authorities. Operational Roles may be added according to the administrative tasks identified.

7.8.3 Usage

The construct Organizational Unit is used in all model phases. Its template shall enable the capturing of all Organization positional information relevant for model-based planning and decision support, monitoring and control of operational processes as well as their execution and its presentation in a form that is understandable by human-and interpretable by digital technology. The template shall

identify all Organizational Units to which this Organizational Unit is assigned and all Organizational Units and Organizational Roles which are part of this Organizational Unit.

NOTE 1 The organizational areas that are depicted by an Organizational Unit can be of very different sizes (e.g. the whole enterprise, a department, a section or a team).

NOTE 2 Organizational structures can describe reporting lines, profit centres, responsibilities of individuals or teams and the relationships between them.

NOTE 3 An Organization Unit authority/responsibility can change from one life cycle phase to another as the enterprise model evolves.

7.8.4 Construct template for Organizational Unit

Table 8 — Organizational Unit template

Header	
Construct label	<Model-unique abbreviation> OU
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. OU_7
Name	[<adjective> <noun>] Name of Organizational Unit, where <adjective> qualifies the Organizational Unit, and <noun> relates to the scope of the Organizational Unit.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having the authority to design or maintain this Organizational Unit.
Body	
Attributes relevant for domain identification and later phases	
Description	<string> Textual description of the purpose and objectives of the Organizational Unit.
Attributes applicable at design specification and later phases	
Organization Position	<string> Textual description of the position of this Organizational Unit in relation to the organizational structure of the enterprise.
Relationships applicable at design specification and later phases	
Operational Authorities and Responsibilities	
Process-related authority/responsibility	[<identifier> '/' <name>]* Process-related Domains, Business Processes, Enterprise Activities, Services and Events for which this Organizational Unit has authority or responsibility or both.
Information-related authority/responsibility	[<identifier> '/' <name>]* Information-related Enterprise Objects, Object Views, Products and Orders for which this Organizational Unit has authority or responsibility or both.
Resource-related authority/responsibility	[<identifier> '/' <name>]* Resource-related Person Profiles, Resources, and Capabilities for which this Organizational Unit has authority or responsibility or both.
Assignments	
Assigned Organization Roles	[<identifier> '/' <name>]+ Organizational Roles that are assigned to this Organizational Unit
Assigned Organizational Units	[<identifier> '/' <name>]+ Organizational Units that are assigned to this Organizational Unit.

Table 8 *(continued)*

Managed by	[<identifier>]* Organizational Unit (s), if any, having authority or responsibility for this Organizational Unit.

NOTE 1 Initially the Design Authority attribute can have no values or a placeholder value, which is resolved to an existing construct before the model is complete.

NOTE 2 The consistency of the Operational Authorities and Responsibilities and the corresponding attributes of the named constructs is the responsibility of the implementation tool.

7.9 Decision Centre

7.9.1 Purpose

The purpose of the Decision Centre construct is to enable the representation of the decisional structure of the enterprise. In terms of organizational entities, a Decision Centre represents a set of decision-making activities that are characterized by having the same time planning parameters and belonging to the same decision function category. Each Decision Centre has its contents described in terms of a decision function category and its relationships to other Decision Centres, which describe relevant decisional and information flows, and to Organizational Units.

7.9.2 Description

A Decision Centre construct shall describe a decision frame with decision objectives, variables, constraints, decision function categories and decision levels. The decision category defines the kinds of decisions with which the decision centre is concerned, e.g. manage products, manage resources, plan production. The decision level represents a decision timeframe that is characterized by decision horizon (time interval) for which a decision is taken, and a decision period (time interval) at the end of which a decision is revised.

The combination of decision level and decision function category define the position of a Decision Centre in the decisional structure.

NOTE B.4.3 contains more details on decision level and decision function category, and on the representation of the decisional structure (see B.4.3.4.1) in a Graphs of Interrelated Results and Activities (GRAI) grid (Figure B.11).

Each Decision Centre shall contain at least one relationship to another Decision Centre and one to an Organizational Unit. The relationships between Decision Centres and with Organizational Roles and Units shall be described by means of associations in the design specification and later phases.

7.9.3 Usage

The construct Decision Centre is used in the concept definition phase and later phases. Its template shall enable

a) the capturing of all decision-related information relevant for model-based planning and decision support, monitoring and control of operational processes, as well as their execution, and

b) its presentation in a form that is understandable by humans and interpretable by digital technology.

7.9.4 Construct template for Decision Centre

Table 9 — Decision Centre template

Header	
Construct label	<Model-unique abbreviation> DC
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. DC_8
Name	<verb> [[<adjective>]? <noun>]? Name of Decision Centre, where <verb> characterizes the imperative nature of the Decision Centre, <adjective> and <noun> are optional qualifier and scope indicators of the Decision Centre.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having the authority to design or maintain this Decision Centre.
Body	
Attributes applicable at concept definition and later phases	
Description	<string> Textual description of the remit of the Decision Centre.
Attributes applicable at requirements definition and later phases	
Decision Frame	<string> | [<identifier> '/' <name>] (i) text listing the objectives to be achieved, the variables involved and the constraints to be satisfied, or (ii) [defined only in a Decision context (see Clause B.4)] reference to a Decision Frame construct (see B.4.3.2) which can apply to several Decision Centres, characterizing the information to be transmitted to and/or from the Decision Centre.
Decision Function Category	<string> | [<identifier> '/' <name>] Text describing the kind of decision activities or Decision Centres handling the same kinds of decision-making activities and concerning the same kinds of subjects.
Decision Level	[<horizon> ':' <period>] Horizon and period, specified in terms of days, weeks or years. *<horizon> specifies the time interval upon which a decision taken stands and <period> specifies the reviewing frequency at the end of which a decision taken needs to be revised.*
Attributes applicable at design specification and later phases	
Organizational Position	<string> Textual description of the position of this Decision Centre in relation to Organizational Units of the enterprise.
Relationships applicable at design specification and later phases	
Operational Applicability	
Process applicability	[<identifier> '/' <name>]* Domains, Business Processes, Services, Enterprise Activities, Decision Activities (only in a Decision View) and Events to which decisions made by this Decision Centre apply.
Information applicability	[<identifier> '/' <name>]* Enterprise Objects, Object Views, Products and Orders to which decisions made by this Decision Centre apply.
Resource applicability	[<identifier> '/' <name>]* Person Profiles, Resources and Capabilities to which decisions made by this Decision Centre apply.
Assignments	

Table 9 *(continued)*

Controlled Decision Centres	[<identifier> '/' <name>]* Decision Centres controlled by this Decision Centre.
Assigned to Decision Centre	[<identifier> '/' <name>] Decision Centre that has responsibility and authority on this Decision Centre.
Assigned to Organization Role	[< identifier> '/' <name>] Organizational Role responsible for the Decision Centre.
Assigned to Organization Unit	[<identifier> '/' <name>]+ Organizational Unit(s) to which this Decision Centre is assigned.

7.10 Role

7.10.1 Purpose

The purpose of the generic (abstract) construct Role is to represent roles played by human or organizational entities in support of or in relation to other constructs such as Business Process, Enterprise Activity, Service and Resource. As an abstract construct, only specializations of Role can have instances (see 8.4).

7.10.2 Description

A Role construct shall describe for early modelling life cycle phases the general or typical or characteristic functions performed by entities, which are given more specificity via specialization at later modelling life cycle phases. The same entity may have different roles in different contexts that may overlap in time and place, and at different times over its life cycle.

EXAMPLE Different roles and contexts include:

— for human resources, an employee can be in turn a subordinate in the structure of a given organization, a manager of other employees in the same structure and an external part-time consultant in the structure of another organization; this person can provide different sets of capabilities when involved in his or her different roles;

— for an organizational entity, an enterprise can be one or several of a Collaborating Partner in a collaboration, a user of a Service concerned with the functionality of a Service, or a Co-provider in resourcing that service.

7.10.3 Usage

The various roles described above are represented by the following specializations of Role in 8.4:

— Organizational Role

— Operational Role

— Person Profile

— Collaborating Partner

— Stakeholder (further specialized into User and Co-provider)

7.10.4 Construct template for Role

Table 10 — Role template

Header	
Construct label	<Model-unique abbreviation> RO

Table 10 *(continued)*

Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5)
Name	[<adjective> <noun> | <noun>] Name of Role, where <adjective> optionally qualifies the Role and <noun> relates to the scope of the Role.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having the authority to design or maintain this Role.
Body	
Attributes applicable at concept definition and later phases	
Description	<string> Textual description of the remit of the Role.
Attributes applicable at requirements definition and later phases	
Attributes	[<attribute_name> [' = ' [<attribute_value>]+]? Elements representing attributes and possibly their values for the entity represented by the Role.
Attributes applicable at design specification and later phases	
Organizational Position	<string> Textual description of the position of this Role in relation to the organizational structure.
Relationships applicable at design specification and later phases	
Assignments	
Assigned to construct	[<identifier> '/' <name>]+ Construct to which this Role is assigned.

7.11 Product

7.11.1 Purpose

The purpose of the Product construct is to represent characteristics of the product that are relevant for its intended user, enterprise or industry sector and also technical information that is relevant for management, administration, planning and control of the desired output or by-product of Business Processes, Services or Enterprise Activities.

7.11.2 Description

A Product construct shall describe the properties and constraints of relevant technical and administrative information of a product and specializations thereof, including relationships to other products (e.g. versions, others). A Product construct is a specialization of an Enterprise Object construct.

By particularization and specialization of attributes, Product subclasses shall be defined to meet the specific requirements of a certain branch of industry or distinct enterprise. Product structures shall be represented by Part-of relationships between well-distinguished Product subclasses.

7.11.3 Usage

The construct Product is used in all model phases. Its template shall enable

a) the capturing of all product-related information relevant to the intended User (see 8.4.5) and as required by management and administration for strategic planning and decision support, monitoring and control of operational processes, and

b) its presentation in a form that is understandable by humans and interpretable by digital technology.

NOTE Depending on the extent of detail, all Product stages relevant to the manufacturing process are described as inputs/outputs of the processes involved and their logical sequence representing the processing of the Product.

7.11.4 Construct template for Product

Table 11 — Product template

Header	
Construct label	<Model-unique abbreviation> PR
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. PR_10
Name	<string> Name of the Product.
Design Authority	[<identifier> '/' <name>] [':' [<identifier> '/' <name>]]? Names of Organizational Unit and possibly Organizational Role, having authority to design or maintain this Product.
Body	
Attributes relevant for domain identification and later phases	
Description	<string> Short textual description.
Nature of Object	[PHYSICAL | INFORMATION] Nature of the entity represented by the Order.
Attributes	[<attribute_name> ['=' [<attribute_value>]*]?]+ Elements representing attributes and possibly values of the Product where — <attribute_name 'location'> indicates location of the product — <attribute_name 'stage'> indicates its state relevant to the manufacturing processing of the Product.
Attributes applicable at requirements definition and later phases	
Constraints	[<constraint>]* Constraints (s) imposed on attributes of the Product.
Integrity Rules	[<integrity rule>]* Integrity rule (s) applicable to attributes of the Product.
Functionality	[<string>] | [Functionality derived construct]* Text describing the functionalities or [defined only in a Servitization specialization (see 8.3)] references to Functionality derived construct (see 8.4.7).
Relationships relevant for domain identification and later phases	
Class Relationships – list of Product relationships in the form:	
Specialization-of	[<identifier> '/' <name>]* Product classes that are a generalization of this Product. *The name of the Product being defined cannot be used in the list.*
Part-of	[<identifier> '/' <name>]* Products in which this Product is involved in an aggregation. *The name of the Product being defined cannot be used in the list.*
Consists-of	[<identifier> '/' <name>]* Products of which this Product is the aggregate. *The name of the Product being defined cannot be used in the list.*
Related-to	[<identifier> '/' <name>]* Enterprise Objects that are related to this Product, where multiplicity [...]* is one of [0..*] (only for early modelling phases) or [1..1] or [1..n] or [m..n].
Relationships applicable at design specification and later phases	

Table 11 *(continued)*

Operational Relationships	
Operation Responsibility	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Operational Role and Organizational Unit respectively, having responsibility for operation of this Product.
Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Role and Organizational Unit respectively, having authority for this Product.

7.12 Order

7.12.1 Purpose

The purpose of an Order construct is to represent the management and administrative order information for planning and control of Business Processes in an enterprise and in particular to describe an instruction from one authority to other authorities for performance of an operation.

7.12.2 Description

An Order construct shall describe what needs to be done, which products and sub-products need to be produced, e.g. Parts list, which resources need to be used, and what purchases need to be made, e.g. Bill of materials. An Order is a specialization of the Enterprise Object construct for which integrity rules are valid for all phases, and which has a distinguishing construct label to reflect its purpose and usage.

The interdependencies of the various kinds of authorization in a company are created through relations based on the Kind of Order attribute. These include:

— the outward-facing Customer order which is the information relevant to the planning, control, and monitoring of business processes for the acceptance, delivery and billing of a customer order;

— the inward-facing Production Order which is the issuing of authoritative commands for the execution of enterprise activities to produce the product;

— a Hybrid order which is part Customer Order and part Production Order.

The Kind of Order attribute is typically a two- or three-word phase which summarizes a distinct sub-class of order requiring context-dependent attributes. A modeller is free to define specializations of Order corresponding to each kind. Because of the variability of order detail between different kinds of order, attributes are not defined in the templates other than as being a list of variable names and values.

Specific states of the Orders trigger Business Processes or Enterprise Activities that transform an Order from an input order state to an output order state, e.g. from 'customer order received' to 'order completed'.

7.12.3 Usage

The construct Order is used in all model phases. Its template shall enable

a) the capturing of all order-related information relevant for strategic planning and decision support, monitoring and control of operational processes, and

b) its presentation in a form that is understandable by humans and interpretable by digital technology.

The Order shall contain all information necessary for the execution of any internal and external enterprise order. Any order shall be executed by the responsible Organizational Unit.

NOTE 1 An association of an Order with an Event is represented by an Object View.

NOTE 2 Part of an Order can be represented by an Object View that is associated with a Start Event.

Within an enterprise, many different kinds of Orders with different associated information may be defined (e.g. customer order, production order, etc.).

EXAMPLE 1 Controlling attributes for a Customer order

— Order Identification: [<name> ',' <administrative identifier>]

— Order Item: [<requested performance> ',' [<other>]*]

— Order Amount: [<order volume> ',' <unit of quantity> ',' <actual state>]

— Order Dates: [<ordered date> ',' <delivery date> ['earliest' | 'latest' | 'actual'] ',' <state >

— Order State: [<scheduled date> ['earliest' | 'latest' | 'actual'] | 'prepared' | 'active' | 'finished' | 'suspended' | 'cancelled']

— Order Priority: <priority ranking>

— Destination: <external User> ',' <delivery address> ',' <internal User/Organization identifier>]

EXAMPLE 2 Controlling attributes for a Production order

— Work plan: [<Production order identification> ',' <Order Amount> ',' <date (earliest start) > ',' <date (latest end) > [<resource>]*]* – a list of production orders together with required resources.

— Work status: [<state name> [<attribute> '=' <value>]*] representing a specific (interim or final) result in the production process which can be related to the orders within the work plan.

EXAMPLE 3 Controlling attributes for a Hybrid order

A Hybrid order will contain elements of both Customer and Production order. For example, a contract manufacturing order with high visibility into the production process will contain some specification as to how that order will be produced or supplied. Attribute details will vary according to the extent of process visibility and the degree of automation and artificial intelligence available from the provider but will include some commercial order attributes and some production order attributes.

7.12.4 Construct template for Order

Table 12 — Order template

Header	
Construct label	<Model-unique abbreviation> OR
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. OR_11
Name	<string> Name of the Order.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having authority to design or maintain this Order.
Body	
Attributes relevant for domain identification and later phases	
Description	<string> Short textual description.
Nature of Object	[PHYSICAL | INFORMATION] Nature of the entity represented by the Order.
Kind of Order	<string> Short phrase describing a particular kind of order used in a specific context that has distinguishable attributes from other kinds.
Attributes applicable at requirements definition and later phases	

Table 12 *(continued)*

Constraints	[<constraint>]* Constraint (s) imposed on attributes of the Order.
Integrity Rules	[<integrity rule>]* Integrity rule (s) applicable to attributes of this Order.
Controlling Attributes	[<attribute_name> ' = ' <attribute_value>]+ Elements representing Order attributes and their values that impose control requirements on Business Processes and/or Enterprise Activities and/or Services.
Relationships relevant for domain identification and later phases	
Class Relationships – list of Order relationships in the form:	
Specialization-of	[<identifier> '/' <name>]* Order (s) that are a generalization of this Order. *The name of the Order being defined cannot be used in the list.*
Part-of	[<identifier> '/' <name>]* Orders in which this Order is involved in an aggregation. *The name of the Order being defined cannot be used in the list.*
Consists-of	[<identifier> '/' <name>]* Orders of which this Order is the aggregate. *The name of the Order being defined cannot be used in the list.*
Related-to	[<identifier> '/' <name>]* Enterprise Objects that are related to this Order (such as Product), qualified by a multiplicity [...]* which is one of [0..*] (only for early modelling phases) or [1..1] or [1..n] or [m..n]. *The name of the Order being defined cannot be used in the list.*
Relationships applicable at requirements definition and later phases	
Associated Event	[<identifier> '/' <name>]? Event signalling the arrival of this Order.
Relationships applicable at design specification and later phases	
Operational Relationships	
Operation Responsibility	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Operational Role and Organizational Unit respectively, having responsibility for operation of this Order.
Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Role and Organizational Unit respectively, having authority for this Order.

7.13 Resource

7.13.1 Purpose

The purpose of the Resource construct is to identify a set of Capabilities provided for a Business Process or Enterprise Activity requiring those Capabilities. This includes all consumables, equipment and Operational Roles needed to operate the equipment.

7.13.2 Description

The construct describes in the form of capabilities and other properties all material and processing aids in an enterprise, such as machining equipment, tools and facilities, equipment for data processing, documents and files containing geometrical, material and informational characteristics, which are required to execute, enable or support the execution of Business Processes or Enterprise Activities in an enterprise. A Resource does not include human resources, which are instead assigned to the resource in the form of an Operational Role.

A Resource is a specialization of Enterprise Object that specializes its properties and adds new properties to describe

— assigned operational roles, and

— provided capabilities.

The resulting construct template has a distinguishing construct label to reflect its purpose and usage.

For each kind of Resource, all relevant qualities and services of that Resource are described in terms of a Capability, related to its ability to complete tasks in the enterprise and its availability for, and constraints on, carrying out those tasks.

Resources may be passive like consumables, fixtures and pallets. Alternatively, they may be active and capable of carrying out operations, often under human direction (e.g. workstations, robots). Active behaviour is enabled by an Operations Set and a Skills Set, which are empty for a Resource that is passive.

The Operations Set contains a list of operations identified in the design specification phase, following from a decomposition of the provided Capabilities of an active Resource, that are matched to the required Capabilities of an Enterprise Activity.

NOTE The control flow of these operations, collectively called activity behaviour, can be defined in an algorithm, which describes the input/output conditions for sequence control of the operations. Unlike the presence of Behavioural Rules in other constructs, the specification of the algorithm is an implementation issue beyond the scope of this document and this document does not specify a language or form of representation for the control flow algorithm.

The Skills Set lists the skills that will be provided by an Operational Role. Each skill is identified by a skill name and described in terms appropriate and understandable to the person providing or entity requiring them; the skill name needs to be unique within the scope in which it is defined and used.

7.13.3 Usage

Resource is used In an Enterprise Activity as an input to identify the required and provided capabilities. Resource outputs contain status information after the execution of a task.

The construct Resource is used in the requirements definition phase and later phases. Its template shall enable

a) the capturing of all resource-related information relevant for planning and decision support, monitoring and control of operational processes, and

b) its presentation in a form that is understandable by humans and interpretable by digital technology.

The description above also implies some subclasses of Resource that are specialized and specified for a particular enterprise.

7.13.4 Construct template for Resource

Table 13 — Resource template

Header	
Construct label	<Model-unique abbreviation> RE
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. RE_12
Name	<string> Name of the Resource.

Table 13 *(continued)*

Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having authority and responsibility to design or maintain this Resource.
Body	
Attributes relevant for domain identification and later phases	
Description	<string> Short textual description.
Attributes applicable at requirements definition and later phases	
Nature of Object	[PHYSICAL | INFORMATION] Nature of the entity represented by the Resource.
Attributes	[<attribute_name> ['= '<attribute_value>] ?]+ Elements representing attributes and possibly their values of the Resource. *These attributes define necessary attribute characteristics other than those that are task-related; as an example, attribute names can be location, when installed, book value. Attributes that are task-related are contained within the associated Capability.*
Constraints	[<constraint>]* Constraint imposed on the attributes of the Resource.
Integrity Rules	[<integrity rule>]* Integrity rule (s) applicable to the attributes of the Resource.
Operations Set	[<operations>]* Operations that will be provided by this Resource. *Each operation is identified by an operation name and its input and output arguments; the operation name needs to be unique within the scope in which it is defined and used. The operations set is empty for passive resources.*
Skills Set	[<skill>]* Skills that will be provided by an Operational Role. *Each skill is identified by a skill name and described in terms appropriate and understandable to the person providing or entity requiring them; the skill name needs to be unique within the scope in which it is defined and used. The skills set is empty for passive resources.*
Relationships applicable at design specification and later phases	
Class Relationships – list of Resource relationships in the form:	
Provided Capabilities	[<identifier> '/' <name>]* Capabilities provided by this Resource.
Operational Role	[<identifier> '/' <name>]* Required Operational Roles.
Specialization-of	[<identifier> '/' <name>]* Resources which are a generalization of this Resource. The name of the Resource being defined cannot be used in the list.
Part-of	[<identifier> '/' <name>]* Resources of which this Resource is involved in an aggregation. The name of the Resource being defined cannot be used in the list.
Consists-of	[<identifier> '/' <name>]* Resources of which this Resource is the aggregate. The name of the Resource being defined cannot be used in the list.
Related-to	[<identifier> '/' <name>]* List of Enterprise Objects that are related to this Resource, qualified by a multiplicity [...]* which is one of [0..*] (only for early modelling phases) or [1..1] or [1..n] or [m..n].
Operational Relationships	
Operation Responsibility	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Operational Role and Organizational Unit respectively, having responsibility for operation of this Resource.

Table 13 *(continued)*

Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Role and Organizational Unit respectively, having authority for this Resource.

7.14 Capability

7.14.1 Purpose

The purpose of the Capability is to represent the capabilities required by an Enterprise Activity (see 7.3) or those provided by a Resource (see 7.13).

7.14.2 Description

A Capability construct shall describe, by means of capability attributes and other included Capabilities, the abilities required to support execution of the task performed by an Enterprise Activity, and, for identified objects, shall describe any processing time restrictions or constraints such as safety and security aspects, certain tooling and working space dimensions, data processing/storing, main memory/storage capacity, etc.

The Capability attributes being provided by a Resource shall describe each Resource-dependent attribute and a value, set of values, or permissible range of values for that attribute.

EXAMPLE Permissible ranges can be maximum/minimum part sizes or weights that can be handled, degree of precision or repeatability, etc.

7.14.3 Usage

The construct Capability is used in the requirements definition phase and later phases. Its template shall enable

a) the capturing of all Enterprise Activity- and Resource-related information relevant for model-based planning and decision support, monitoring and control of operational processes, and

b) its presentation in a form that is understandable by humans and interpretable by digital technology.

The Capability required by an Enterprise Activity shall be matched by a Resource provided Capability.

7.14.4 Construct template for Capability

Table 14 — Capability template

Header	
Construct label	<Model-unique abbreviation> CA
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. CA_13
Name	<string> Name of this Capability.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having authority to design or maintain this Capability.
Body	
Attributes applicable at requirements definition and later phases	
Description	<string> Short textual description.

Table 14 *(continued)*

Function-related Attributes	[<capability attribute>]* Function-related attributes of this Capability that are available to provide the Required Capabilities of Enterprise Activities.
Constraints	[<constraint>]* Constraint (s) imposed on task-related attributes.
Integrity Rules	[<integrity rule>]* Integrity rule (s) applicable to task-related attributes.
Attributes applicable at design specification and later phases	
Performance-related Attributes	[<capability attribute>]* Capability attribute (s) identifying and constraining performance-related attributes.
Operation-related Attributes	[<capability attribute>]* Capability attribute identifying and constraining Enterprise Activity-related attributes.
Relationships applicable at requirements definition and later phases	
Capabilities Included	[<identifier> '/' <name>]* Included Capabilities.
Where_required	[<identifier> '/' <name>]+ Enterprise Activities to which this Capability is provided.
Where_provided	[<identifier> '/' <name>]+ Resources that are required for this Capability.
Relationships applicable at design specification and later phases	
Operational Relationships	
Operation Responsibility	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Operational Role and Organizational Unit respectively, having responsibility for operation of this Capability.
Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Role and Organizational Unit respectively, having authority for this Capability.

7.15 Performance Indicator

7.15.1 Purpose

The purpose of the Performance Indicator is to represent measurements of performance that are relevant to the management of the many enterprise value creation processes and the continuous improvement of those processes. An appropriate set of performance indicators, derived from measurement of operation performance, aids in determining if the processes complete as planned and the goals are realized as expected.

Performance measures are particularly meaningful for the realization of operational performance improvement.

7.15.2 Description

A Performance Indicator construct shall describe a measurement specification pertaining to operational performance of an enterprise entity. A performance indicator quantifies data to measure the extent of achievement of an enterprise objective.

Within an enterprise, the various operational areas, such as sales, manufacturing, engineering, marketing, and other business support functions, have different sets of performance indicators.

EXAMPLE Examples of specific constructs for performance measurement are Business Processes, Enterprise Activities, Organizational Units, Orders, Products, Services or Person Profiles.

7.15.3 Usage

The monitoring of performance is specific to identified objectives of the enterprise, and performance indicators are most useful when their values aid the identification of trends relative to certain operational objectives. Taken together, often combining more specific performance indicators, various performance indicators provide information for monitoring the realization of enterprise business objectives. A good performance indicator has certain criteria that ensure its usefulness in achieving various goals in enterprise operation.

NOTE ISO 22400-1 provides criteria for key performance indicators in the domain of manufacturing operations management.

Monitoring the trend for a Performance Indicator construct over time is essential to effective analysis of performance and process improvement. The trend of an indicator of performance can result from the formula calculation, e.g. using exponential smoothing, or result from a means provided by the implementation, in which case trend determination is beyond the scope of this document.

The Performance Indicator construct is used in the requirements definition phase and later phases.

7.15.4 Construct template for Performance Indicator

NOTE This template includes properties derived from the ISO 22400 KPI template.

Table 15 — Performance Indicator template

Header	
Construct label	<Model-unique abbreviation> PI
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. PI_14
Name	<string> Name of the Performance Indicator.
Design Authority	<identifier> '/' <name>] ':' [<identifier> '/' <name> Organizational Unit and Organizational Role respectively, having authority to design or maintain this Performance Indicator.
Body	
Attributes relevant for domain identification and later phases	
Description	<string> Short textual description.
Concern	<string> Text defining the kind of concern addressed by this Performance Indicator. These can include value to the enterprise, user satisfaction, quality, time, risk, speed, accuracy and traceability.
Attributes applicable at requirements definition and later phases	
Objective	[<string>]+ Targeted objectives for this Performance Indicator.
Attributes applicable at design specification and later phases	
Confidentiality	<restriction> Text defining a restriction of visibility for Performance Indicator results (e.g. company confidential, only board members, public).
Formula	<formula> Mathematical formula for calculating Performance Indicator value using measurements and possibly other Performance Indicators.
Indicator Value	[<number> | <string>]?

Table 15 *(continued)*

Unit of measure	<string> Text defining basic unit or dimension in which the Performance Indicator is expressed.
Range	['lower'|'upper'] Text defining the lower and upper logical limits of the Performance Indicator.
Preferred trend	['Higher'|'Lower'] Preferred improvement direction, whichever is better.
Timing	['real-time']? ['on-demand']? ['periodically']? A Performance Indicator should be calculated in one or more of: — real-time - after each new data acquisition event — on demand - after a specific data selection request — periodically - done at a certain interval, e.g. once per day
Relationships applicable at design specification and later phases	
Assesses	[<identifier> '/' <name>]+ Kind of construct that the Performance Indicator assesses.
Stakeholder	[<string> | <Stakeholder derived construct>]+ List of individuals with an interest in this Performance Indicator, or [defined only in a Servitization context, (see 8.3)] references to Stakeholder derived construct instances (see 7.4.4). *These can include Operators, Supervisors and Management.*
Information required to build the indicator	[<identifier> '/' <name>]* List of required Object Views for this Performance Indicator, where multiplicity [...]* is one of [0..*] or [1..*].
Affected constructs	[<identifier> '/' <name>]+ List of constructs that can be affected by the Performance Indicator results.
Operational Relationships	
Operation Responsibility	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Operational Role and Organizational Unit respectively, having responsibility for operation of this Performance Indicator.
Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Role and Organizational Unit respectively, having authority for this Performance Indicator.

7.16 Template attributes for core constructs

Figure 7 expands Figure 4 to show construct attributes. Here and elsewhere, construct relationship properties that are listed in construct templates are shown as UML associations, not attributes.

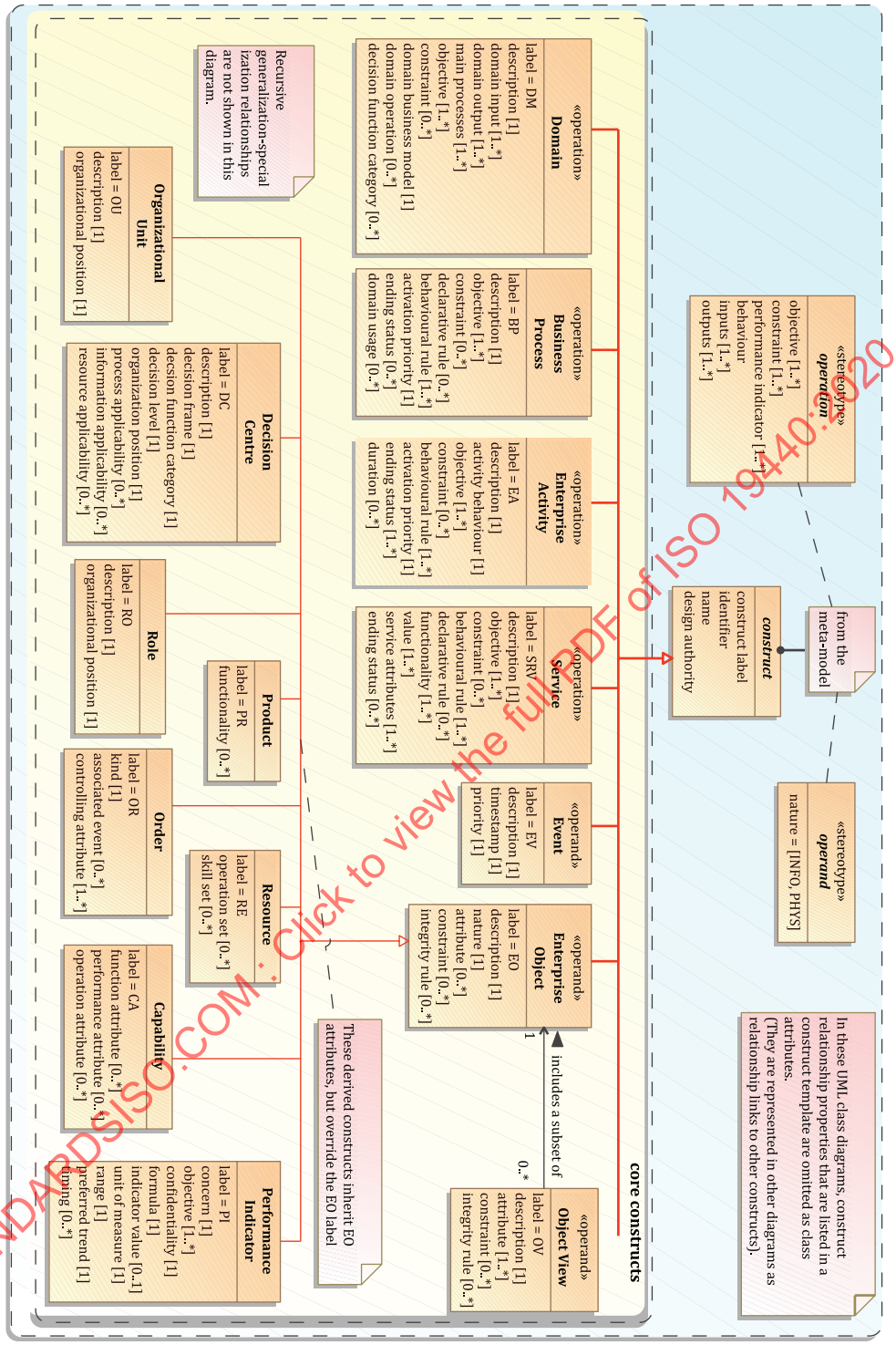


Figure 7 — Core constructs showing attributes

8 Derived constructs

8.1 Overview

Derived constructs are formed by specializing core constructs, adding new properties and possibly modifying or constraining existing properties. The set of derived constructs specified in 8.2 are for manufacturing-oriented enterprises and the set specified in 8.3 are for service-oriented enterprises.

A user of this document may extend these constructs and any other constructs to derive new or specialized constructs for other kinds of enterprise models. Annex B illustrates several more derived constructs.

8.2 Derivations for a manufacturing-oriented enterprise

8.2.1 Purpose

The purpose of a manufacturing-oriented derivation is to enable explicit specification of responsibilities for and authorizations to modify other modelling constructs. The assignment of Organizational Roles, Operational Roles and Person Profiles, each of which is a specialization of the core Role construct, achieves this specification in 8.4.1, 8.4.2 and 8.4.3 respectively.

8.2.2 Use of core constructs

The following core constructs are inherited without modification or re-interpretation:

— Domain,

— Business Process,

— Enterprise Activity,

— Event,

— Enterprise Object,

— Object View,

— Organizational Unit,

— Decision Centre,

— Role,

— Product,

— Order,

— Resource,

— Capability, and

— Performance Indicator.

For the purpose of modelling a manufacturing-oriented enterprise, for each kind of Resource, all relevant qualities and services of that Resource shall be described in terms of

a) Capability (see 7.14), related to its ability to complete functional activities in the enterprise and its availability for, and constraints on, carrying out those tasks,

b) Operational Role (see 8.4.2), defined as a specialization of Role (see 7.10) to identify human capabilities required to operate this Resource, and

c) corresponding operations for acquiring or preserving the ability and the availability of a) and b) respectively, e.g. preparation, provision, servicing, as well as the logical sequence of these operations.

The Resource construct is extended by adding at the design specification and later phases a new relationship property as the first item of Class Relationships as:

Attributes applicable at design specification and later phases	
Provided Operational Roles	[<identifier> '/' <name>]* of the Operational Roles providing human capabilities to operate this Resource

The Organizational Unit construct is also extended by inserting at the design specification and later phases new assignment relationship properties as the first item of Assignments. This becomes:

Assignments	
Assigned Operational Roles	[<identifier> '/' <name>]+ identifying the Operational Roles that are assigned to this Organizational Unit
Assigned Organization Roles	[<identifier> '/' <name>]+ identifying the Organizational Roles that are assigned to this Organizational Unit

8.2.3 Manufacturing derivations and core constructs

The main relationships between these core constructs and the manufacturing-derived constructs are illustrated in Figure 8, while the detailed relationships between manufacturing- derived constructs are illustrated in Clause B.3, which also includes function, information, resource and organization views.

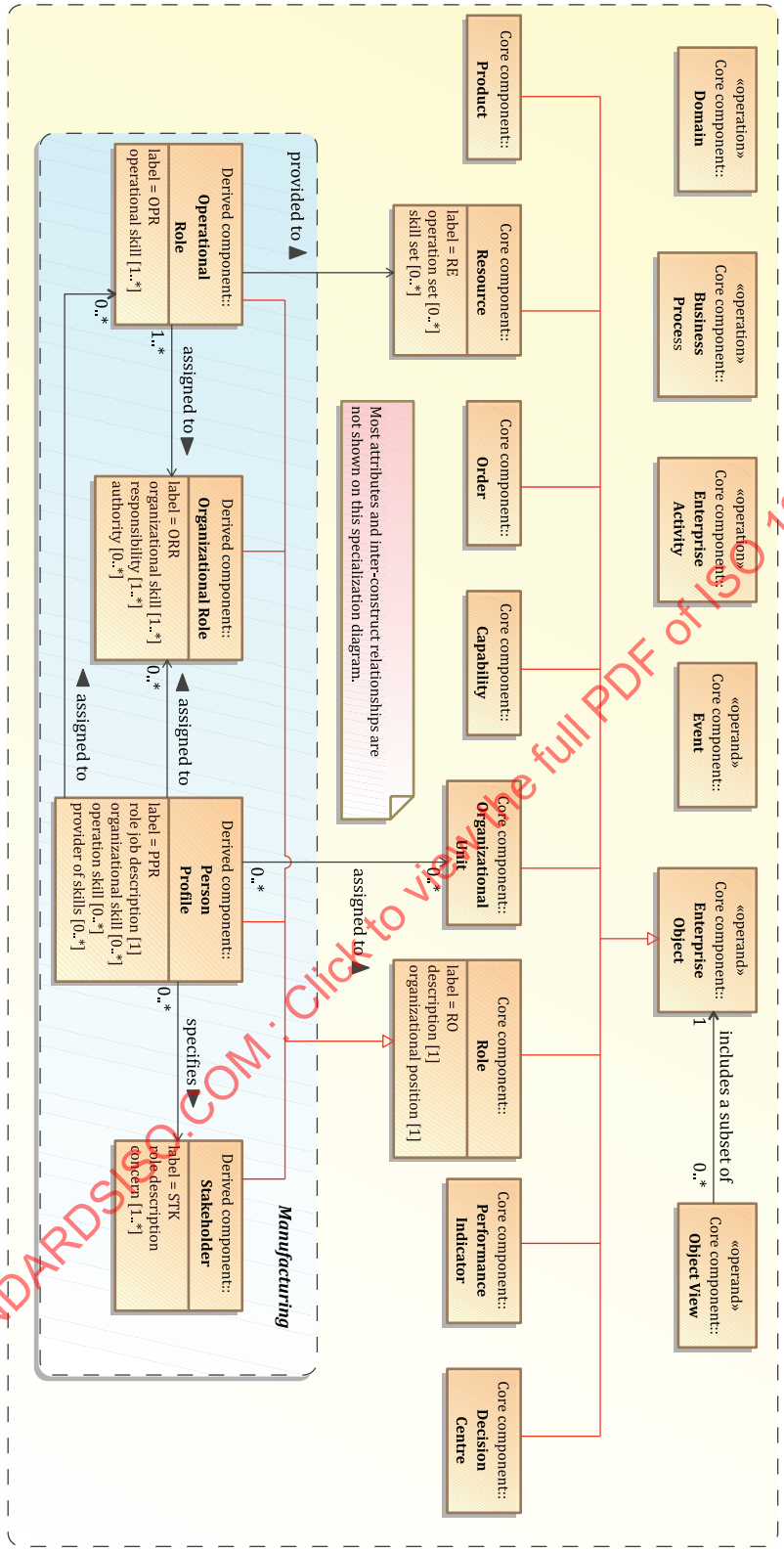


Figure 8 — Manufacturing derivations and core constructs

8.3 Derivations for a service-oriented enterprise

8.3.1 Purpose

A service system provides functions that are utilities to employ in fulfilling user's needs. This benefits the user by increasing equipment availability and reducing time to repair because of pre-existing contractual arrangements, increasing equipment functionality and capability through on-line or on-site upgrading, and allowing increased service integration with customers' internal systems and processes by formalizing some service system interfaces as accessible Application Programme Interfaces (APIs).

For an equipment supplier, a service system can allow access to new markets that can be serviced remotely, reduce costs through formal service contracts with standard conditions and limited variability (e.g. response times), and reduce system development costs through commonality of service system processes across multiple products and kinds of customer.

A service system model benefits the provider by providing a common modelling language for the design and operation of the system, improving quality in the design process and reducing costs in system operation, fostering the development of more compatible ICT products in enterprise service modelling and reducing problems in the interoperation of such ICT products through interoperable software from co-operating organizations without the need for significant (re) engineering.

Performance indicators, which are related to a set of decisions that control the service system, can evaluate service effectiveness. The decisions are made with respect to a decision structure, the consistency and coherence of which should be analysed.

8.3.2 Service system suppliers

A service-oriented enterprise considers three categories of suppliers:

— small or medium-sized enterprises (SMEs) or large companies active in model-based servitization or in the process of designing business models that include servitization aspects;

— national/regional projects or IT consultancies willing to apply this document in their project/domain;

— large industrial enterprises, which need business models aimed at offering IT assets to other large industrial partners who are looking for standards-based service innovation.

8.3.3 Use of core constructs

For the purpose of modelling a service-oriented enterprise, the following core constructs are inherited without modification or re-interpretation:

— Business Process,

— Enterprise Activity,

— Decision Centre,

— Service,

— Event,

— Enterprise Object,

— Object View,

— Organizational Unit,

— Decision Centre,

— Role,

— Product,

— Order,

— Resource,

— Capability, and

— Performance Indicator.

In a service-oriented enterprise, a Role construct specializes into a Stakeholder construct (see 8.4.4), with further specializations for a User construct (see 8.4.5) and a Co-provider construct (see 8.4.6), and Enterprise Object is specialized into a Functionality construct (see 8.4.7). Properties for these new constructs are inherited or defined in the concept definition modelling life cycle phase and specified in the requirements definition modelling life cycle phase.

8.3.4 Service derivations and core constructs

The relationships between the core constructs and the service-oriented constructs are illustrated in Figure 9.

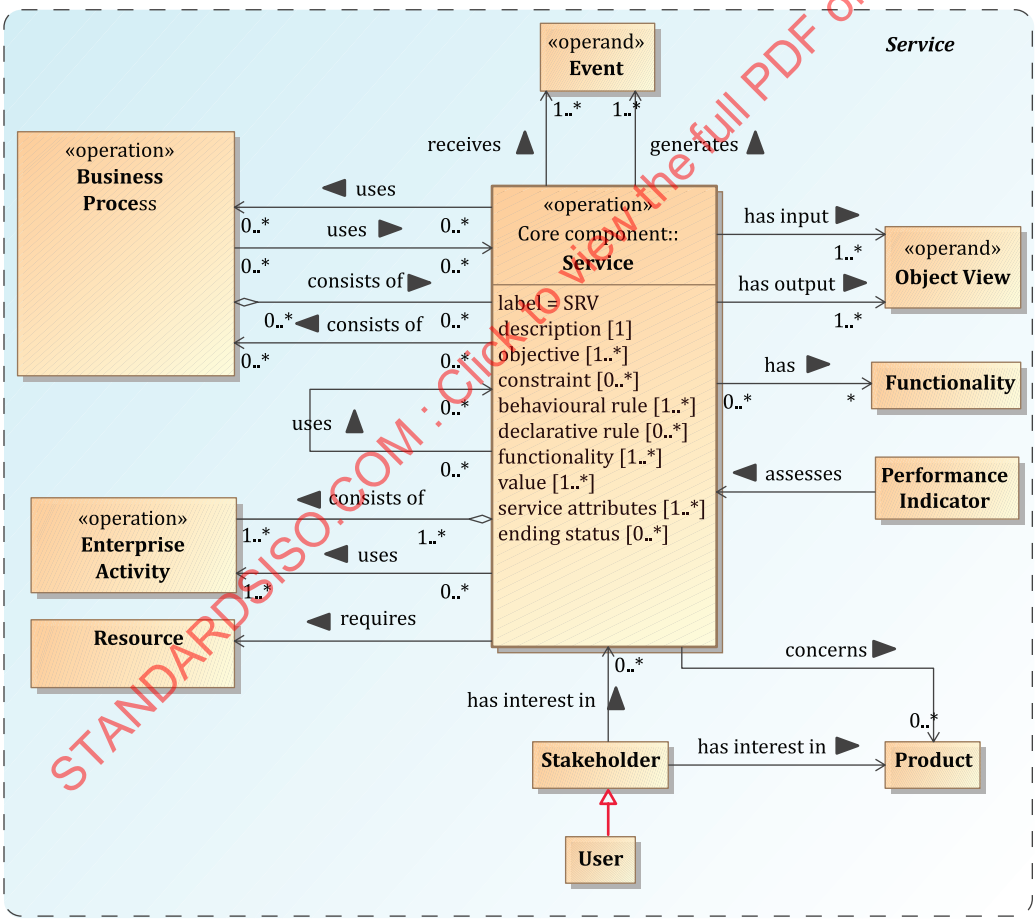


Figure 9 — Service relationships

8.4 Derived constructs

8.4.1 Organizational Role

8.4.1.1 Purpose

The purpose of an Organizational Role construct, which is a specialization of Role (see 7.10), is to represent the skills profile required to serve the defined organizational responsibilities, and to fulfil those responsibilities. A skill profile shall be a list of preferred or user-defined human organizational skills.

8.4.1.2 Description

An Organizational Role construct shall describe, within a given administrative structure of an enterprise, the organizationally relevant human skills and responsibilities required and provided to perform those organizational responsibilities that are assigned to the Organizational Role. Organizational skills shall be described in terms appropriate to the user of those skills and understandable to the person providing or entity requiring them.

Each Organizational Role shall contain at least one relationship to an Organizational Unit.

8.4.1.3 Usage

The Organizational Role construct is used in requirements definition and later phases. Its template shall enable

a) the capturing of all organizational responsibility-related information relevant for model-based planning and decision support, monitoring and control of operational processes, and

b) its presentation in a form that is understandable by humans and interpretable by digital technology.

In the design specification and later modelling life phases, an assignment shall describe the relationships between Organizational Roles and Organizational Units.

8.4.1.4 Construct template for Organizational Role

Table 16 — Organizational Role template

Header	
Construct label	<Model-unique abbreviation> ORR
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. ORR_1
Name	[<adjective> <noun>] Name of Organizational Role, where the adjective qualifies the Organizational Role and the noun relates to the scope of the Organizational Role.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having the authority to design or maintain this Organizational Role.
Body	
Attributes applicable at requirements definition and later phases	
Role job description	<string> Brief textual description of organizational tasks.
Organizational Skills	[<skill>]+ Skills which are required for organizational, decision-making and problem-solving responsibilities.

Table 16 *(continued)*

Responsibilities	[<responsibility>]+ Responsibilities given to Organizational Role as a textual description of organizational, decision-making or problem-solving responsibilities.
Authorities	[<authority>]* Text describing authorities given to this Organizational Role.
Relationships applicable at requirements definition and later phases	
Specialization-of	[<identifier> '/' <name>]* Organizational Roles that are a generalization of this Organizational Role. *The name of the Organizational Role being defined cannot be used in the list.*
Generalization-of	[<identifier>'/'<name>]* Organizational Roles that are a specialization of this Organizational Role.
Relationships applicable at design specification and later phases	
Assignments	
Assigned to Organizational Unit	[<identifier> '/' <name>] Organizational Unit to which this Organizational Role has been assigned.
Assigned to Decision Centre	[<identifier> '/' <name>]? Decision Centre to which this Organizational Role has been assigned.

8.4.2 Operational Role

8.4.2.1 Purpose

The purpose of an Operational Role construct, which is a specialization of Role, is to represent the skill profile required and provided to undertake the defined operational tasks. A skill profile shall be a list of predefined or user-defined operational skills, described in terms appropriate and understandable to the person providing or entity requiring them.

8.4.2.2 Description

An Operational Role construct describes the relevant skills and responsibilities required and provided to perform the operational tasks that are assigned to the Operational Role.

Each Operational Role shall contain at least one relationship to a using entity, i.e. Capability or Organizational Unit. The relationships between Operational Roles and other entities shall be described by means of an assignment relationship in the requirements definition and later phases.

8.4.2.3 Usage

The construct Operational Role is used in requirements definition and later modelling life cycle phases. Its template shall enable

a) the capturing of all operational task-related information relevant for the execution of operational processes, and

b) its presentation in a form that is understandable by humans and interpretable by digital technology.

The template shall identify all Capabilities and Organizational Units to which this Operational Role is assigned.

8.4.2.4 Construct template for Operational Role

Table 17 — Operational Role template

Header	
Construct label	<Model-unique abbreviation> OPR
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. OPR_2
Name	[<adjective>]? <noun> Name of Operational Role, where <adjective> optionally qualifies the Operational Role and <noun> relates to the scope of the Operational Role.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having the authority to design or maintain this Operational Role.
Body	
Attributes applicable at requirements definition and later phases	
Role Job Description	<string> Brief textual description of operational tasks.
Operational Skills	[<skill>]+ Skills which are needed for the operational tasks.
Relationships applicable at requirements definition and later phases	
Specialization-of	[<identifier> '/' <name>]* Operational Roles that are a generalization of this Operational Role. *The name of the Operational Role being defined cannot be used in the list.*
Generalization-of	[<identified.'/'<name>]* Operational Roles that are a specialization of this Operational Role.
Relationships applicable at design specification and later phases	
Assignments	
Assigned to Organizational Unit	[<identifier> '/' <name>] Organizational Unit to which this Operational Role has been assigned.
Assigned to Capability	[<identifier> '/' <name>]+ Capabilities to which this Operational Role has been assigned.

8.4.3 Person Profile

8.4.3.1 Purpose

The purpose of the Person Profile construct, which is a specialization of Role (see 5.5), is to represent the skill profiles available to serve the assigned organizational and operational roles and to fulfil the responsibilities associated with those roles.

8.4.3.2 Description

A Person Profile construct describes the skills that are actually or potentially relevant for the work of Organizational Units (see 7.8) and Enterprise Activities (see 7.3). A skill profile shall be a list of predefined or user-defined organizational and operational skills, described in terms appropriate to the user of those skills and understandable to the person providing them.

Each Person Profile construct shall contain at least one relationship to an Organizational Unit construct and to an Organizational Role construct or Operational Role (see 8.4.2) construct. The relationships between a Person Profile construct and Organizational Unit constructs and Operational Role constructs shall be described by means of an assignment relationship.

8.4.3.3 Usage

The Person Profile construct is used in design specification and later modelling life cycle phases. Its template shall enable

a) the capturing of all skill-related information relevant for model-based planning and decision support, monitoring, control and execution of organizational and operational processes, and

b) its presentation in a form that is understandable by humans and interpretable by digital technology.

8.4.3.4 Construct template for Person Profile

Table 18 — Person Profile template

Header	
Construct label	<Model-unique abbreviation> PPR
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. PPR-3.1
Name	[<adjective>]? <noun> Name of Person Profile, where <adjective> optionally qualifies the Person Profile and <noun> relates to the scope of the Person Profile.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having the authority to design or maintain this Person Profile.
Body	
Attributes applicable at requirements definition and later phases	
Employee Profile relevant for the employment in the enterprise	
Role Job Description	<string> Textual description of organizational, decision-making, problem-solving or operational tasks.
Organization-related Skills	[<skill>]* Skill (s) that is (are) provided for organizational, decision-making and problem-solving tasks.
Operation-related Skills	[<skill>]* Skill (s) that is (are) that are provided for operational tasks.
Provider of Skills	[<identifier> '/' <name>]* Provider, where the <identifier> and <name> specifies the person (s) providing the skill.
Relationships applicable at design specification and later phases	
Assignments	
Assigned to Organizational Unit	[<identifier> '/' <name>] Organizational Unit that has responsibility and authority on this Person Profile.
Assigned to Organizational Role	[<identifier> '/' <name>]* Organizational Roles to which this Person Profile is assigned.
Assigned to Operational Role	[<identifier> '/' <name>]* Operational Roles for which this Person Profile provides skills.

8.4.4 Stakeholder

8.4.4.1 Purpose

The purpose of the Stakeholder construct, which is a specialization of Role (see 7.10), is to represent an entity that has concerns about some other entity, but not necessarily a responsibility for that entity's design or provision.

8.4.4.2 Description

A Stakeholder construct describes a person, group, or organization that has direct or indirect legitimate interest in a service system, a product or performance indicator results. A stakeholder can represent a supplier, an employee of the enterprise, a local or national community, a retailer, an investor, etc.

8.4.4.3 Usage

The Stakeholder construct is used in design specification and later modelling life cycle phases. Its template shall enable

a) the capturing of all concern-, constraint- or performance-related information relevant to the Stakeholder's needs and interests, and

b) its presentation in a form that is understandable by humans and interpretable by digital technology.

8.4.4.4 Construct template for Stakeholder

Table 19 — Stakeholder template

Header	
Construct label	<Model-unique abbreviation> STK
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. STK_4
Name	<string> Name of the Stakeholder.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having the authority to design or maintain this Stakeholder.
Body	
Attributes applicable at requirements definition and later phases	
Role Description	<string> Short textual description of the Stakeholder's role or position.
Concern	[<string>]+ Short textual descriptions of issues, interests, requirements etc.
Constraint	[<constraint>]* Constraints imposed on this Stakeholder.
Relationships applicable at requirements definition and later phases	
Person Profile	[<identifier> '/' <name>]* Person Profiles specifying the stakeholder role.
Entities of interest	[<identifier> '/' <name>]* Products, Services and Performance Indicators with which the Stakeholder is concerned.

EXAMPLE In the template, Person Profile construct name label can include, among others, sponsor, engineer, provider, person in charge of maintenance.

8.4.5 User

8.4.5.1 Purpose

The purpose of the User construct is to represent the interests, and characteristics of an entity intended to use a product or service.

8.4.5.2 Description

A User construct describes entities that will utilize the product or consume the service. The User construct is a specialization of a Stakeholder construct (see 8.4.4), which is itself a specialization of a Role construct (see 7.10). The Characterization attribute shall not include personal information that is protected by privacy laws and attributable to a named individual.

8.4.5.3 Usage

The User construct information may be used to specify the service or product and the system that will provide the service.

8.4.5.4 Construct template for User

Table 20 — User template

Header	
Construct label	<Model-unique abbreviation> USER
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. USER_5
Name	<string> Name of User or type of User.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having the authority to design or maintain this User construct.
Body	
Attributes applicable at requirements definition and later phases	
Category	<string> | [<identifier> '/' <name>]* Occupational category or Person Profile construct specifying the User role.
Characterization	[<string>]* Typical user characteristics affecting the suitability or otherwise of a Product or Service.
Intended frequency of service use	<string> Short textual description.
Requirement	<string> Short textual description.
Constraint	<string> Short textual description.
Relationships applicable at requirements definition and later phases	
Entities of interest	[<identifier> '/' <name>]* Products, Services and Performance Indicators with which the User is concerned.

NOTE For the Category attribute, the modeller can consider using the International Standard Classification of Occupations (ISCO), available at https://www.ilo.org/public/english/bureau/stat/isco/isco08/index.htm.

8.4.6 Co-provider

8.4.6.1 Purpose

The purpose of the Co-provider construct is to represent its role in cooperating with other Stakeholders.

8.4.6.2 Description

A Co-provider construct describes a person group or organization associated with another entity in providing the service. A Co-provider is part of the service system. In a virtual manufacturing enterprise, a Co-provider has a very strong relationships with the service system. The Co-provider construct is a specialization of a Stakeholder (see 8.4.4) construct which is itself a specialization of Role (see 5.5).

8.4.6.3 Usage

The Co-provider construct is used in design specification and later modelling life cycle phases.

8.4.6.4 Construct template for Co-provider

Table 21 — Co-provider template

Header	
Construct label	<Model-unique abbreviation> COPR
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. COPR_6
Name	<string> Name of the Co-provider.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having the authority to design or maintain this Co-provider.
Body	
Attributes applicable at requirements definition and later phases	
Category	<string> Short textual description of the kind of Co-provider.
Competence	[<string>]+ Short textual description of the competence(s) of Co-provider.
Concern	[<string>]* Short textual description of issues and requirements, etc.
Constraint	[<constraint>]* Constraints imposed on this Co-provider.
Contribution	[<contribution>]* Short textual description of the contribution(s) of Co-provider.
Relationships applicable at requirements definition and later phases	
Services of interest	[<identifier> '/' <name>]* Services in which the Co-provider is interested or involved (described by Service construct).

8.4.7 Functionality

8.4.7.1 Purpose

The purpose of the Functionality construct is to represent the characterization of some aspect of what a product or service can or cannot do for a user. These aspects should reflect the reasons why a user

is interested in a product or a service. Aspects may include technical specification or constraints, but not technical functionalities, i.e. the technical means or details regarding the way a functionality is provided.

8.4.7.2 Description

A Functionality construct describes the behaviour of a product or service in terms of the functions it is expected to perform. A Functionality construct is a specialization of an Enterprise Object (see 7.6).

8.4.7.3 Usage

A Functionality construct can be further decomposed into sub-functions. Functionality can serve as the basis of a dialogue between producer and Stakeholder (see 8.4.4) or User (see 8.4.5) on the identification of new user needs and product or service development. Functionality construct properties are first specified in the requirements definition modelling life cycle phase.

8.4.7.4 Construct template for Functionality

Table 22 — Functionality template

Header	
Construct label	<Model-unique abbreviation> FUNC
Identifier	<Model-unique string> Modeller-assigned alphanumeric code in the form of a construct label, followed by an extended alphanumeric string (see 5.5), e.g. FUNC_7
Name	<string> Name of the Functionality.
Design Authority	[<identifier> '/' <name>] ':' [<identifier> '/' <name>] Organizational Unit and Organizational Role respectively, having the authority to design or maintain this Functionality.
Body	
Attributes applicable at requirements definition and later phases	
Category	[<string>]+ Kind of functionality. *Examples can include information processing, material handling, transportation.*
Capabilities/Limitations	[<string>]+ Short textual description of capabilities and limitations.
Decomposition	[sub-function]* List of sub-functions.
Relationships applicable at requirements definition and later phases	
Applies to	[<identifier> '/' <name>]* Products and Services having this Functionality.

NOTE For the Category attribute, the modeller can consider using the International Standard Industrial Classification of All Economic Activities (ISIC), available at https://unstats.un.org/unsd/publication/seriesM/seriesm_4rev4e.pdf.

Annex A
(normative)

Behavioural rules — Detailed description and syntax

A.1 Process behaviour

A.1.1 Enterprise process architecture

This document does not prescribe an enterprise process architecture, but there are at least four different kinds of relationship for the invocation of processes:

a) delegation, where a delegated process is expected to finish or fail and return control with an appropriate ending status;

b) partition, where several processes are involved in a fan-out, fan-in, collective finish or failed relationship;

c) chaining, where one process passes control to another, etc., until the end of the chain when control passes back to the chain originator or some overall scheduler;

d) networking, where the relationships between the co-operating processes are simply invocations, e.g. a collaboration composed on-the-fly to meet a specific demand or project requirement.

A.1.2 Constraining process behavioural rules

The general case is illustrated in Figure A.1 where P, P1 (part of P) and Q are process instances:

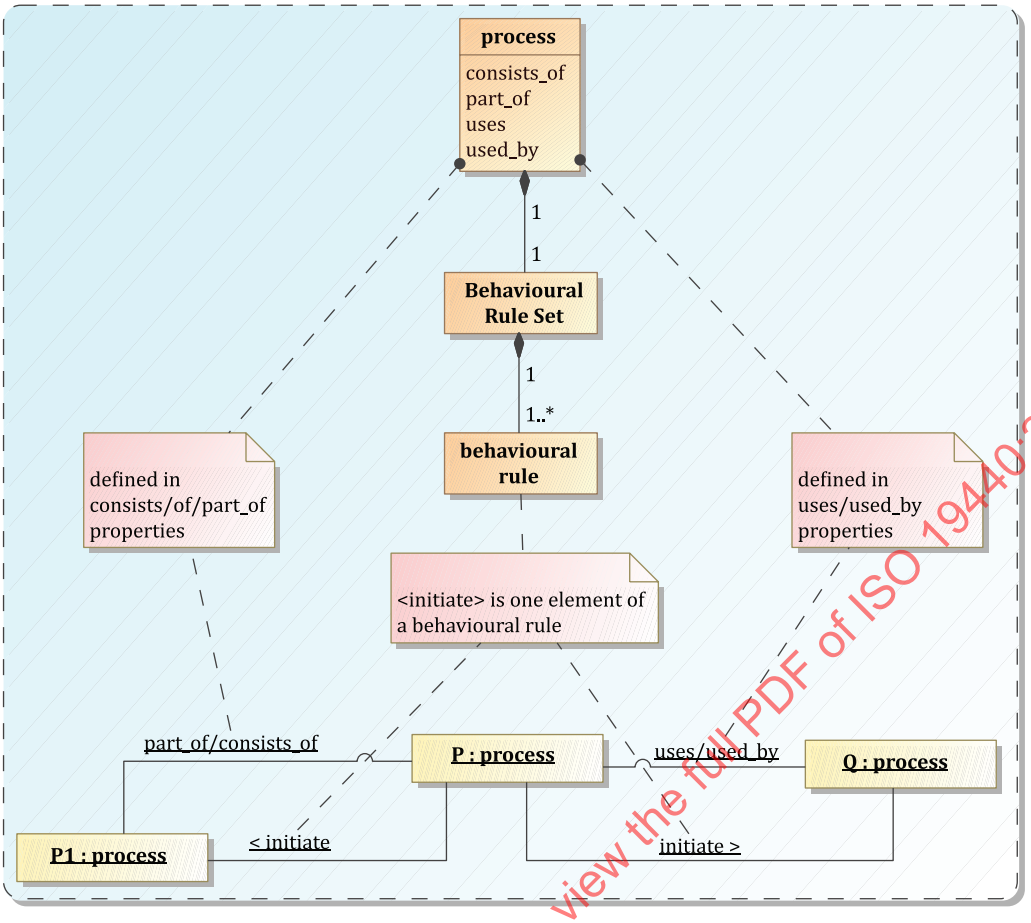


Figure A.1 — Inter-process instance relationships

In order to assist the modeller in avoiding problems, it is necessary to impose some constraints as follows.

a) A behavioural rule in a process shall refer to only those processes that are parts of the same process or are listed in a 'uses' relationship.

NOTE This is not incompatible with 7.1.4 NOTE 2 specifying <origin> for input events and <destination> for output events, because those relationships are not part of a behavioural rule set.

b) The behavioural rule set (BRS) of a process P that consists of other processes P_1, P_2, ... P_n shall completely prescribe the possible process initiations between its part processes P_1, P_2, ... P_n. A part process P_i that consists of other sub-processes shall similarly prescribe their behaviour in its own BRS, not the BRS of P_i's parent process.

c) A process may also in its BRS initiate other processes that are not part of it, either directly via complementary uses/used_by relationships or indirectly via output_event/input_event relationships, subject to the following constraints:

1) the uses relationship list in a process shall not include that process or parts of that process;

2) the used_by relationship in a process shall list all other processes that may use that process;

3) a behavioural rule in a process that generates an event shall separately list that event in an output event property;

4) a process that may respond to an event occurrence shall separately list that event in an input event property.

These constraints enable the modeller (usually assisted by a computer tool) to trace inter-process invocations and identify unhandled events, possible conflicts and infinite loops. They do not prevent possible process deadlocks arising from process competition for shared resources.

The consists of and uses relationships between Business Process, Services and Enterprise Activities are illustrated in Figure A.2 (the dual part-of and used-by relationships are not shown).

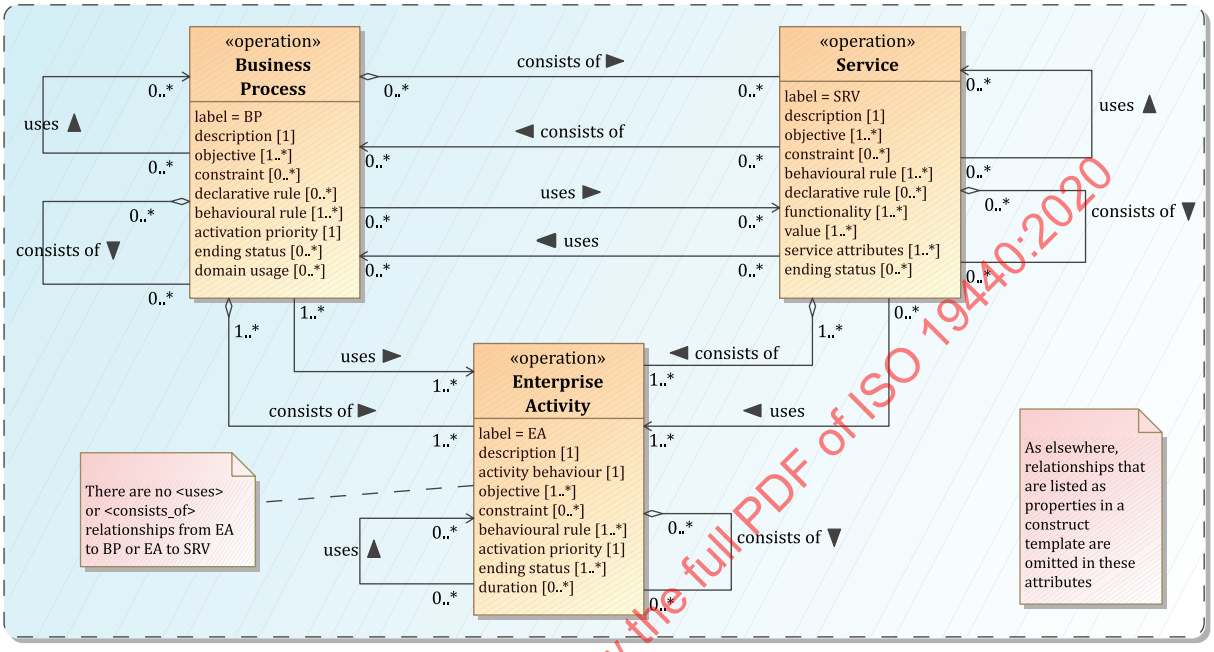


Figure A.2 — Relationships between process constructs

A modeller can decide whether to have:

— a strict process hierarchy comprised of only 'consists-of'/'part-of' relationships and so constraining behaviour because all BRSs will be on nodes of one hierarchy tree and scopes in that BRS will be limited to sub-trees of that node, or

— a looser network structure comprised of only 'uses'/'used-by' relationships and requiring attention to process interactions through modelling (e.g. OPM, Petrinets, …) to detect deadlocks, loops, etc., or

— a mixed hierarchy/heterarchy structure with local and networked scopes that are clearly delineated by the two different kinds of 'consists-of'/'part-of' and 'uses'/'used-by' relationships which should reduce the amount of confusion and unexpected behaviour but still require attention to process interactions.

However, the syntax and semantics of the behavioural rules do not guarantee qualities of acyclicity, predictability or avoidance of deadlocks (e.g. from competition for resources). Nor do they address issues of parallelism caused by the occurrence of multiple events that initiate the same process, each requiring a significant period of time for the consequential chain of activities to complete (e.g. the delay between ordering and delivering a product or service). This means that the overall process behaviour and the obligation to avoid process loops is the responsibility of the designer of the behavioural rules set, while the requirement to avoid deadlocks and handle near-concurrent requests and long-lived process threads are implementation issues requiring techniques already used by computer operating systems.

This document does not specify a mechanism for interpreting the behavioural rules at run-time.

A.2 Overview

A.2.1 Graphical Illustration

Figure A.3 illustrates the syntax for behavioural rules.

NOTE 1 This class model is not part of the UML model for constructs used in the rest of this document.

NOTE 2 To improve visual traceability, four classes (condition, action, single condition, single action) appear in more than one place, but each is defined (elaborated) only once.

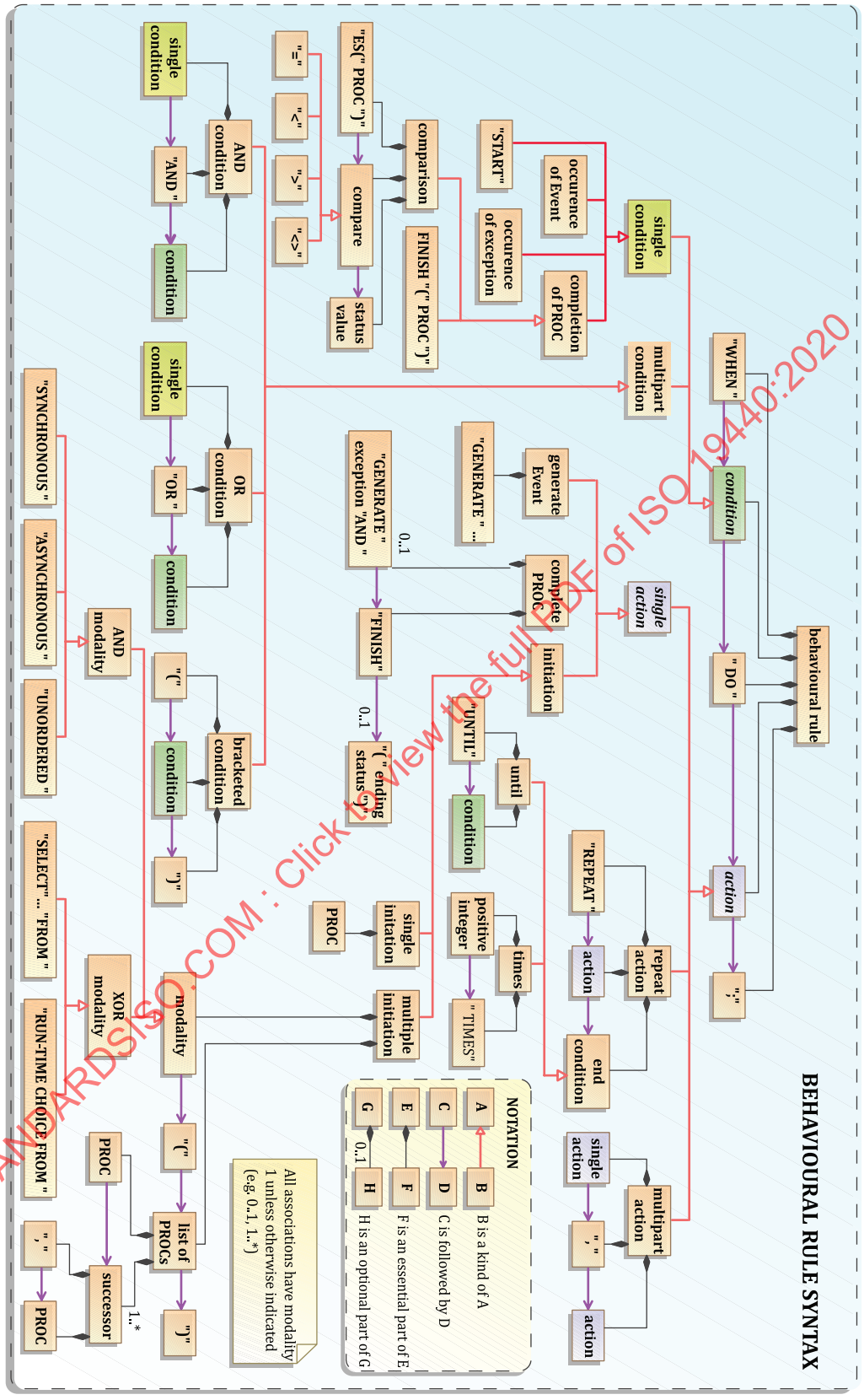


Figure A.3 — Syntax for behavioural rules

A.2.2 Textual description

This clause describes the rules introduced in 7.2.5 in textual form with examples. The formal syntax for behavioural rules is contained in Clause A.3.

In the following, reserved words are denoted totally in upper case.

NOTE 1 The reserved keywords occurring in this annex are AND, ANY, ASYNCHRONOUS, BPEA, DO, ES, EXCEPTION, FINISH, FROM, GENERATE, OR, REPEAT, RUN-TIME CHOICE, SELECT, START, SYNCHRONOUS, TIMES, UNTIL, UNORDERED, WHEN and XOR. Each term can be replaced by an equivalent term in all capital letters provided that is understandable to the human reader.

A Business Process (BP) or Service (SRV) or Enterprise Activity (EA) is denoted by '<PROC>' with instances <PROC1>, <PROC2>, etc.

Behavioural rules are defined with the following general form:

WHEN condition DO action;

where

a) the ';' character terminates a behavioural rule;

b) condition is one or more of the following:

   1) the occurrence of one or more designated events: as further described in A.2.3.1, these include normal events (represented by Events and START events for Business Processes or Enterprise Activities) and exceptions;

   2) the completion of one or more <PROC>s that have been previously initiated as a consequence of an action of the behavioural rules of the <PROC> that contains those rules (see A.2.4.1);

   3) a multi-part condition, consisting of either

      i) two or more conditions separated by the reserved word AND, in which case the multi-part condition is defined to occur if all of those conditions occur, or

      ii) two or more conditions separated by the reserved word OR, in which case the multi-part condition is defined to occur if any of those conditions occur.

c) action is one or more of the following:

   1) initiate one or more <PROC>s (A.2.4.1);

   2) complete a <PROC> with or without raising an exception (A.2.4.2);

   3) generate an Event (A.2.4.3);

   4) repeat a specific action (looping) (A.2.4.4).

NOTE 2 In the case where both AND and OR separators are used in one multi-part condition [item c) above], parentheses ' (' and ') ' or equivalent grouping symbols are used to group conditions so as to resolve ambiguities.

EXAMPLE The condition START AND (Event1 OR Event2) occurs if START occurs and either Event1 or Event2 occurs, while (START AND Event1) OR Event2 occurs if START and Event1 both occur or Event2 occurs.

NOTE 3 The syntax does not of itself prevent infinite loops, as in a <PROC 1> that has WHEN START DO <PROC 1> as a behavioural rule, or more indirectly WHEN START DO <PROC 2> where <PROC 2> (or another Business Process invoked directly or indirectly by <PROC 2>) also contains a rule WHEN START DO <PROC 1>. Implementations can impose restrictions, e.g. that no <PROC> can initiate itself or a another <PROC> that initiates this containing <PROC> either directly or indirectly, but such restrictions are implementation matters outside the scope of this document.

NOTE 4 As with initiation of a <PROC>, implementations can impose restrictions in order to avoid infinite loops, e.g. that no <PROC> responds (via a condition in a behavioural rule) to an Event that it generated, either directly or indirectly, but such restrictions are implementation matters outside the scope of this document.

A multi-part action is represented as a list of actions separated by commas, which, if more than one <PROC> needs to be initiated, shall be preceded by a reserved word (indicating either an 'and' modality [see A.2.3.1 b)] or an 'exclusive or modality' [see A.2.3.1 c)]) denoting constraints on their otherwise parallel initiation (see A.2.3.1). Commas may be replaced by the (semantically equivalent) reserved words AND (for an and modality) or XOR (for an exclusive or modality).

NOTE 5 Behavioural rules do not have associated identifiers since they are attributes within a <PROC>.

NOTE 6 The two kinds of condition and four kinds of action, together with their various qualifying modalities, can be used in certain combinations to express various patterns of sequencing relationships between Business Processes or Enterprise Activities, or some combination of both (synchronous/asynchronous; fan-out/fan-in, etc.). Some example patterns, and the corresponding conditions and actions for these, are summarized in Table A.1. Other kinds of patterns, conditions or actions can exist or be developed later.

Table A.1 — Example behavioural rule patterns, with corresponding conditions and actions

Typical patterns (non-normative)	Condition	Possible actions (and modalities) (see A.2.3)
Start-up process initiation – the unconditional initiation of one or more <PROC> on initiation of this Business Process	START [see A.2.3.1 b)]	Initiate action
Event initiation – the initiation of one or more <PROC> following the occurrence of a designated event	Event [see A.2.3.1 a)]	Initiate action
Conditional – causing the conditional initiation of a successor <PROC>, dependent on the ending status of its predecessor	Action status-completion [see A.2.3.2 a)]	Initiate action
Forced – causing the initiation of a successor <PROC> regardless of the ending status of the predecessor	Action on- completion [see A.2.3.2 b)]	Initiate action
Parallel spawning – causing the concurrent, parallel initiation of two or more successor <PROC>	Any condition	Initiate action: AND-branching (synchronous, asynchronous)
Unordered set – where a set of <PROC> needs to be executed in total, but the order of execution is not known beforehand and is determined at run-time)	Any condition	Initiate action: AND-branching (unordered)
Rendezvous – synchronizing the end of spawned <PROC>	AND-completion [see A.2.3.2 c)]	Any action
Run-time choice – making an exclusive choice among several possible successor <PROC>	Any condition	Initiate action, XOR-branching (select, run-time choice)
Alternate completion– on completion of any one of a group of Business Processes or Enterprise Activities	OR-completion [see A.2.3.2 d)]	Any action except repeat iteratively
Loop – repeating the initiation of a successor <PROC> until a given condition holds or for a defined number of iterations	loop completion: REPEAT (UNTIL, TIMES)	Initiate action, loop
Business Process completion – indicating the end of execution of a set of behavioural rules and the termination of its containing Business Process	normal completion: with ending status	Complete/ terminate Business Process: FINISH

A.2.3 Conditions

A.2.3.1 Event

An event condition is one of two kinds of conditions used in behavioural rules. The occurrence of a designated event is represented by the event name or identifier. Event conditions have three forms

a) normal events generated during the normal operation of a <PROC> and represented by an Event construct.

b) start, the fact of initiation of <PROC>, denoted by the reserved word START. This is used for example to initiate set-up and preparatory actions within each <PROC>. If a behavioural rule set contains more than one behavioural rule, the rules containing START conditions are evaluated first in the same order as they are defined, followed by all other rules, again in the same order as they are defined.

NOTE 1 START can be combined with one or more normal events in a behavioural rule. In that case, the behavioural rule will have no effect (no preparatory actions will occur) unless both START and all the designated events have occurred.

c) exceptions raised as the result of a termination (abnormal ending) of a previously initiated <PROC>, or by a special external mechanism such as a watchdog timer, denoted by name of the exception. Exceptions have two forms

1) designated, if an exception is raised of the designated kind, represented by EXCEPTION exception name. The exception name shall be unique within the scope in which it is defined and used.

NOTE 2 The reserved word EXCEPTION is optional. It is intended as an aid to the human reader.

2) default (intended as a catch-all), if an exception of any kind is raised that has not been otherwise designated and represented by EXCEPTION ANY.

A.2.3.2 Action completion

An action completion is one of two kinds of conditions used in behavioural rules. Its occurrence signifies the completion of one or more designated <PROC> (each having a corresponding ending status). When a <PROC> has been initiated several times within a loop action as described in A.2.4.4, only the last termination is considered as an action completion. Action completion has four forms.

a) On-status condition – the action is initiated dependent on a comparison of the ending status of an Enterprise Activity with a designated value and represented by one of

— ES (<PROC>x) = status value

— ES (<PROC>x) < status value

— ES (<PROC>x) > status value

— ES (<PROC>x) <> status value

where <PROC> is a designated <PROC>, ES (<PROC>) denotes the ending status of that <PROC>, and status value is any constant or expression that can be evaluated and tested for equality, less than, greater than or inequality relative to the ending status.

b) On-completion condition – the action is initiated when a <PROC> completes (regardless of its ending status) provided there is no on-status comparison that has evaluated to true, represented by

FINISH (<PROC>)

and is equivalent to ES (<PROC>) = ANY

c) AND-completion condition (rendezvous) – all of a list of designated <PROC> complete, represented by

completion condition <PROC1> AND completion condition <PROC2> … AND completion condition <PROCn>

where each completion condition <PROC>i (i=1..n) is either an on-status condition or an on-completion condition as defined in A.2.3.2 a) and b) above.

d) OR-completion condition – at least one <PROC> in a list of designated <PROC> completes, represented by

completion condition <PROC1> OR completion condition <PROC2> … OR completion condition <PROCn>

where each completion condition <PROC>i (i=1..n) is either an on-status condition or an on-completion condition as defined in A.2.3.2 a) and b) above.

A.2.4 Actions

A.2.4.1 Initiate action

The initiate action signifies that one or more <PROC> is to be initiated by a condition occurrence as defined in A.2.2. The initiate action models the fact that a <PROC> can be initiated only on completion of its predecessor [see A.2.3.2] or on the occurrence of a start event [see A.2.3.1 b)], or the occurrence of a normal event [see A.2.3.1 a)], or a raised exception [see A.2.3.1 c)].

EXAMPLE The initiation of a <PROC> can be dependent on the availability of a necessary Resource or Event, even though its predecessor has terminated and enabled it to proceed.

The initiate action has three forms

a) single action – a single designated <PROC> is initiated, represented simply by its name, as in the behavioural rule

WHEN START DO prepare_Workstation;

b) AND-branching – enabling the activation of multiple successor <PROC> by one or more predecessors. The ordering of successor activation is determined by its <and modality>, which may be synchronous (all starting at the same time), asynchronous (activated in parallel but possibly starting at different times) or unordered (the sequence of activation is not known and will be determined at run-time, possibly subject to temporal constraints). The action for AND process-branching is represented by

<and modality> <PROC1>, <PROC2>,..., <PROCn>

where <and modality> is one of

SYNCHRONOUS or ASYNCHRONOUS or UNORDERED

The commas in the list of <PROC> may be replaced by the reserved word AND.

c) XOR-branching – enabling the activation of one and only one successor <PROC> by one (or more) predecessor(s). The selection of successor activation is determined by its exclusive or modality, which may be case-determined (indexed or determined by the run-time value of some variable), or run-time choice (determined by a decision made at run-time). The action for XOR-process-branching is represented by

<exclusive or modality> <PROC1>, <PROC2>,..., <PROCn>

where <exclusive or modality> is one of

— SELECT jth item FROM <PROC1>, <PROC2>,..., <PROCn> (used for selection of <PROCj> by a run-time value of j)

— RUN-TIME CHOICE FROM <PROC1>, <PROC2>,..., <PROCn> (used for selection by a run-time decision)

The commas in the list of <PROC> may be replaced by the reserved word XOR.

A.2.4.2 Complete action

The complete action is enabled by any completion condition as defined in A.2.3.2 and terminates the <PROC>. It has two forms

a) normal completion – the <PROC> terminates normally, represented by the keyword FINISH and optionally indicating an ending-status. It is represented by

FINISH ;

or

FINISH (<status value>) ;

b) exception termination – the <PROC> raises an exception (usually as the result of a designated or default exception condition) and thereupon terminates. It is represented by

GENERATE <exception name> AND FINISH ;

A.2.4.3 Generate event action

The generate event action enables a <PROC> to generate an Event. It is represented by

GENERATE <event name> ;

A.2.4.4 Loop action

The loop action signifies the repetitive initiation of some action. A loop action has two forms

a) repeat conditionally – a test condition is evaluated after each repetition of the action to be repeated. If the condition is true the loop exits, otherwise the action is repeated, and the condition is checked again. Conditional looping is represented by

REPEAT <action> UNTIL test condition ;

where <action> is either initiate action (single action, AND-branching or XOR-branching) or a further loop action, and test condition is an action completion as defined in A.2.2.2.

NOTE REPEAT <PROC> UNTIL ES (<PROC>) = ANY is semantically equivalent to <PROC>.

b) repeat iteratively – providing for a fixed number of repetitions of an action and represented by

REPEAT <action> <loop count> TIMES ;

where <action> is either initiate action (single action, AND-branching or XOR-branching) or a further loop action, and <loop count> is an expression, which either evaluates to a positive integer that then determines the number of repetitions or, if it does not so evaluate, the action is not initiated.

A.3 Formal syntax in Extended Backus-Naur Form (EBNF)

The following syntax uses the notation of EBNF as described in ISO/IEC 14977. EBNF syntactic elements are typeset in Courier font to distinguish them from normal text and to ensure that quotes are shown

correctly. The normal character representing each operator of Extended BNF and its implied precedence is (highest precedence at the top)

* repetition-symbol

- except-symbol

, concatenate-symbol

| definition-separator-symbol

= defining-symbol

; terminator-symbol

The normal precedence is over-ridden by the following bracket pairs:

'	first-quote-symbol	first-quote-symbol	'
"	second-quote-symbol	second-quote-symbol	"
(*	start-comment-symbol	end-comment-symbol	*)
(	start-group-symbol	end-group-symbol	)
[	start-option-symbol	end-option-symbol	]
{	start-repeat-symbol	end-repeat-symbol	}
?	special-sequence-symbol	special-sequence-symbol	?

NOTE 1 The first-quote-symbol and second-quote-symbol are straight quotes, not curly quotes.

NOTE 2 A space character enclosed in quotes as in ' ' denotes that a literal space character is required, otherwise space characters and line endings (so-called white space) have no significance other than delimiting reserved words and meta-identifiers. Each meta-identifier is written as one or more words joined together by hyphens. A meta-identifier can occur on both the left and right sides of a rule, so enabling recursion.

EXAMPLE

```
list = item | item AND list ;  (* uses recursion to specify a non-empty sequence of item
AND item AND item AND item, etc. *)

(* Behavioural rule set definitions start here. *)

(* The following base declarations define positive-integers and certain behavioural rule
strings *)

non-zero-integer = '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9' ;

decimal-digit = '0' | non-zero-integer ;

positive-integer = non-zero-integer {decimal-digit} ;

(* used in XOR-modality and loop *)

upper-case-letter = 'A' | 'B' | 'C' | 'D' | 'E' | 'F' | 'G' | 'H' | 'I' | 'J' | 'K' | 'L'
| 'M' | 'N' | 'O' | 'P' | 'Q' | 'R' | 'S' | 'T' | 'U' | 'V' | 'W' | 'X' | 'Y' | 'Z' ;

lower-case-letter = 'a' | 'b' | 'c' | 'd' | 'e' | 'f' | 'g' | 'h' | 'i' | 'j' | 'k' | 'l'
| 'm' | 'n' | 'o' | 'p' | 'q' | 'r' | 's' | 't' | 'u' | 'v' | 'w' | 'x' | 'y' | 'z' ;

letter = upper-case-letter | lower-case-letter ;

string-character = letter | decimal-digit | '_' | '-' | '/' | ' ' ;

(* note that a string-character can be a space *)
```

```
name = letter {string-character} ;

(* a name in a behavioural rule is an identifying label or variable. It shall start with a
letter, can contain spaces, etc. *)

proc = name ;

(* identify a Business Process or Enterprise Activity by its name *)

ev = name ;

(* identify an Event by its name *)

status-constant = name ;

(* a literal constant that can be compared with Ending Status *)


(* The following identify reserved words for behavioural-rules: *)

(* Reserved words in behavioural rules are delimited by white space *)

AND = ? logical and ? ;

ANY = ? equivalent to a wildcard (a condition which compares ANY with any value is always
true) ? ;

ASYNCHRONOUS = ? activated in parallel but possibly starting at different times ? ;

DO = ? initiate action if the controlling condition occurs ? ;

ES = ? Ending status of the <PROC> ? ;

(* ES (proc) shall evaluate to 0-argument predicate *)

EXCEPTION = ? signal abnormal behaviour ? ;

FINISH = ? signal completion of the <PROC> ? ;

FROM = ? select an identified item from a list ? ;

GENERATE = ? cause an Event or exception to occur ? ;

OR = ? logical or ? ;

REPEAT = ? iterate ? ;

RUN-TIME-CHOICE = ? select an item from a list at the time when a decision is needed ? ;

SELECT = ? identify which list element needs to be selected ? ;

START = ? condition that occurs when a <PROC> is initiated ? ;

SYNCHRONOUS = ? all starting at the same time ? ;

TIMES = ? identify preceding integer as the number of repetitions ? ;

UNORDERED = ? the sequence of activation is not known and will be determined at run-time,
possibly subject to temporal constraints ? ;

UNTIL = ? check for action completion ? ;

WHEN = ? wait for the occurrence of a (possibly multi-part) condition?
;

XOR = ? logical exclusive or ? ;

(* The preliminaries end here. *)

(* Define behavioural-rule-set for <PROC> constructs: *)
```

```
behavioural-rule-set = behavioural-rule {behavioural-rule} ;

behavioural-rule = WHEN condition DO action ';' ;

condition = condition-term

              | condition-term {AND condition-term}

              | condition-term {OR condition-term} ;

condition-term = single-condition

              | ' (' condition ') ' ;

single-condition = event-occurrence

              | action-completion ;

event-occurrence = event

              | start

              | exception ;

event = ev ;

(* the named Event needs to be included in the list of Event Inputs in the <PROC> that
contains this behavioural-rule *)

start = START ;

exception = designated

              | default ;

designated = [EXCEPTION] exception-name ;

exception-name = name ;

default = EXCEPTION ANY ;

action-completion = on-completion

              | and-completion

              | or-completion ;

on-completion = on-status

              | FINISH ' (' proc ') ' ;

on-status = ES ' (' proc ') ' compare status-value ;

compare = '=' | '<' | '>' | '<>' ;

status-value = status-constant

              | name

              | ANY   ;

(* status-value is a constant or expression that evaluates to a name. This grammar
reflects the ambiguity of the language in this respect; status-constant and name are
different semantics, but both are a sequence of string-characters;

ES (proc) compare ANY is always true if <PROC> is complete *)

and-completion = on-completion {AND on-completion} ;

or-completion = on-completion {OR on-completion } ;
```

```
action = initiate
              | complete
              | generate
              | loop ;
initiate = single-action
              | and-branching
              | xor-branching ;
single-action = proc ;
and-branching = and-modality and-action-set ;
and-modality = SYNCHRONOUS | ASYCHRONOUS | UNORDERED ;
and-action-set = single-action {',' single-action}
             | single-action {AND single-action} ;
xor-branching = xor-modality xor-action-set ;
xor-modality = RUN-TIME-CHOICE
              | SELECT nth-action FROM ;
xor-action-set = single-action {',' single-action}
              | single-action { XOR single-action} ;
nth-action = positive-integer ;
(* nth-action is an expression that should evaluate to a positive-integer <= number of
items in the xor-action-set, otherwise an exception is raised *)
complete = [raise-exception] normal ;
raise-exception = GENERATE exception-name AND ;
normal = FINISH {' (' status-value ') '} ;
generate = GENERATE event ;
loop = conditional-repeat
              | iterative-repeat ;
conditional-repeat = REPEAT action UNTIL action-completion ;
iterative-repeat = REPEAT action loop-count TIMES ;
loop-count = positive-integer ;
(* loop-count should be an expression that evaluates to a positive-integer or zero (in
which case the action is not executed), or if it does not so evaluate, an exception needs
to be raised *)
(* Behavioural rule set definitions end here. *)
```

Annex B
(informative)

Model views

B.1 Overview

This annex contains expanded versions of Clause 5 figures. It shows how a model for a manufacturing-oriented enterprise can be analysed in terms of Function, Information, Resource and Organizational model views and defines further models (and additional constructs) for Decision and Collaboration Views. Whereas the enterprise model represents all aspects and with sufficient detail for computer supported enterprise monitoring and control, the model view presents only a selection of those aspects, in order to focus only on details relevant to the model user's particular needs.

Model Views are an open-ended set. The modeller can extend the views defined here to express additional views, for example economic or value-added views.

B.2 Expanded figures for core and derived construct

B.2.1 Core construct relationships

Figure B.1 expands Figure 4 to show construct relationships.

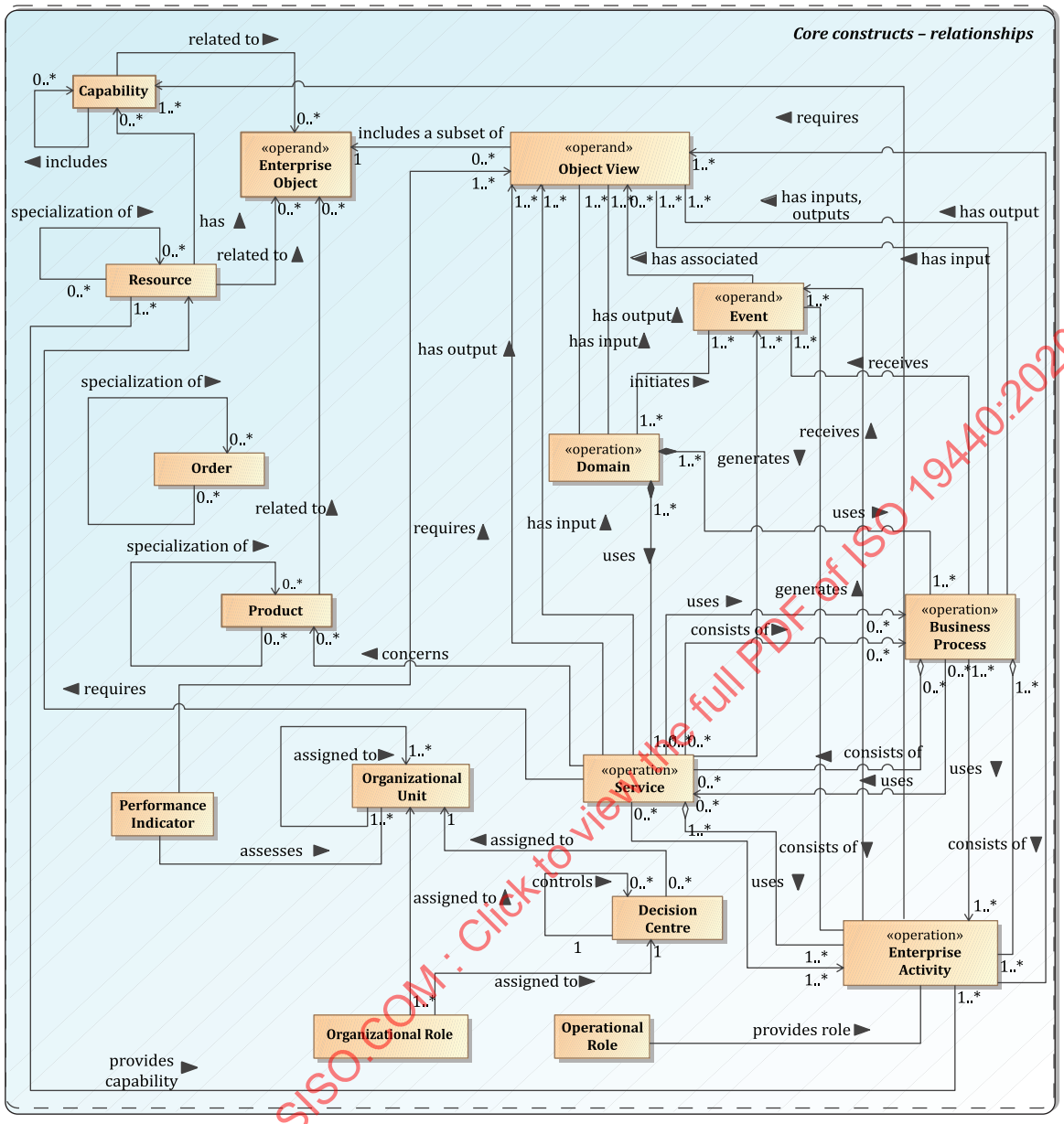


Figure B.1 — Core construct relationships

Further figures in this annex contain the same Figure B.1 relationships, but in more detail (also containing the specializations of 7.16) and organized by Function, Information, Resource and Organization Views.

B.2.2 Derived constructs with attributes

Figure B.2 expands Figure 6 to show attributes for derived constructs.

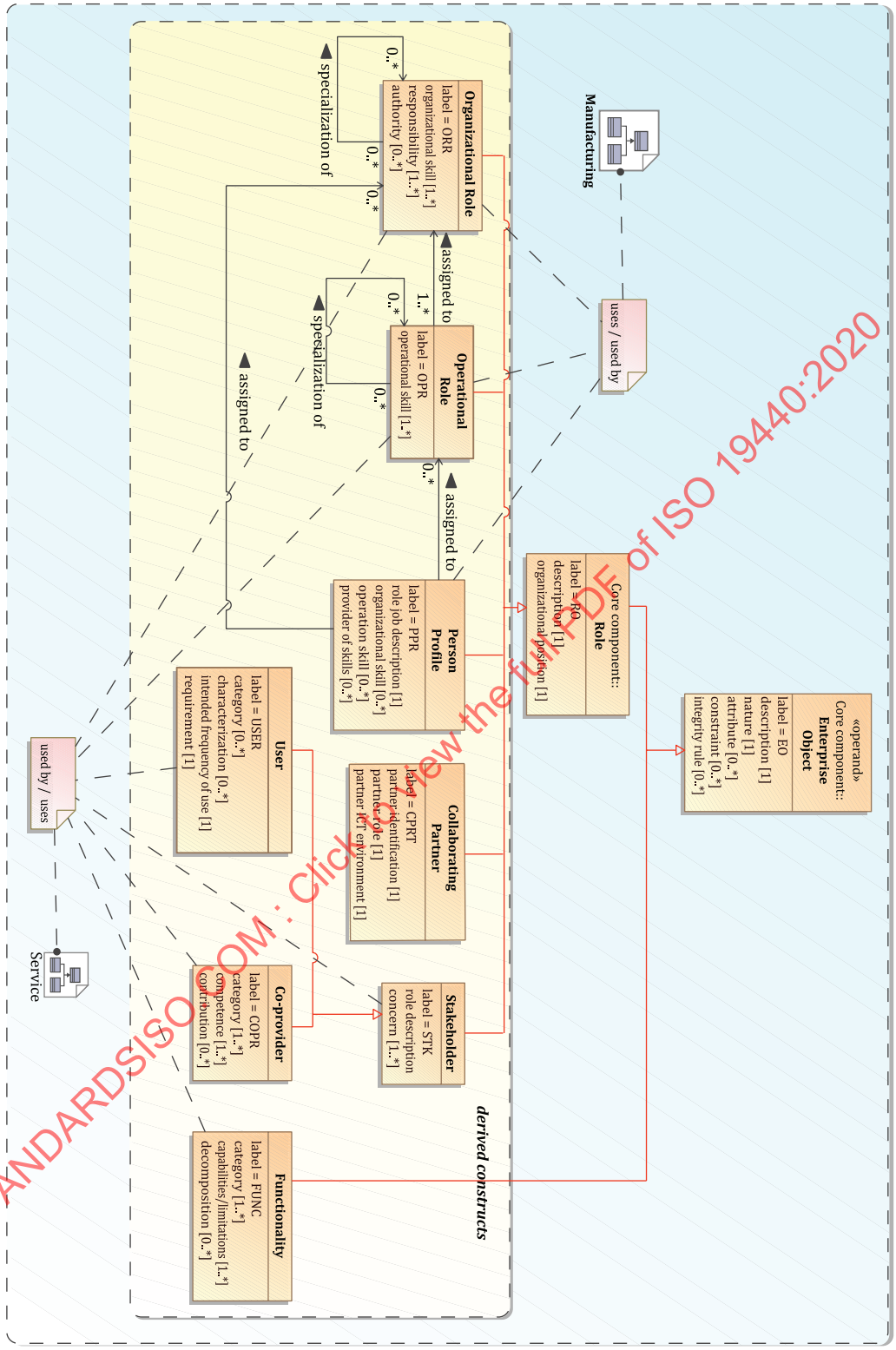


Figure B.2 — Derived constructs with attributes

B.3 Manufacturing-oriented enterprise Views

B.3.1 Overview

The conceptual structure set out in Clause 6 provides an overview of the relationships and attributes of the core constructs of Clause 7 and the specializations of 7.16. The main relationships between those constructs and the manufacturing specializations are shown in 8.2. Figure B.3 expands Figure 8 to include attributes.